

**cyan**

[www.cyantechnology.com](http://www.cyantechnology.com)



ESSENTIAL TEAMWORK



FOR WINNING DESIGNS

**eCOG1X Development Kit  
Version 1.4**

**Cyan Technology**

**eCOG1X Development Kit**

**User Manual**

**V1.4**

**23 January 2008**

## Confidential and Proprietary Information

© Cyan Technology Ltd., 2007-2008

This document contains confidential and proprietary information of Cyan Technology Ltd. and is protected by copyright laws. Its receipt or possession does not convey any rights to reproduce, manufacture, use or sell anything based on information contained within this document.

Cyan Technology™, the Cyan Technology logo and Max-eICE™ are trademarks of Cyan Holdings Ltd. CyanIDE® and eCOG® are registered trademarks of Cyan Holdings Ltd. Cyan Technology Ltd. recognises other brand and product names as trademarks or registered trademarks of their respective holders.

Any product described in this document is subject to continuous developments and improvements. All particulars of the product and its use contained in this document are given by Cyan Technology Ltd. in good faith. However, all warranties implied or expressed, including but not limited to implied warranties of merchantability, or fitness for purpose, are excluded.

This document is intended only to assist the reader in the use of the product. Cyan Technology Ltd. shall not be liable for any loss or damage arising from the use of any information in this guide, any error or omission in such information, or any incorrect use of the product.

This product is not designed or intended to be used for on-line control of aircraft, aircraft navigation or communications systems or in air traffic control applications or in the design, construction, operation or maintenance of any nuclear facility, or for any medical use related to either life support equipment or any other life-critical application. Cyan Technology Ltd. specifically disclaims any express or implied warranty of fitness for any or all of such uses. Ask your sales representative for details.



## Revision History

Version	Date	Notes
V1.0	05/04/2007	First release.
V1.1	16/04/2007	Changed J2 to P3 in section 2.1. Added more information on USB eICE adaptors.
V1.2	17/04/2007	Corrected error in J29 port connections.
V1.3	19/10/2007	Corrected error in J29 port connections for pins 12, 14.
V1.4	23/01/2008	Changed front page.

# Contents

List of Tables . . . . .	v
1 Introduction . . . . .	1
1.1 Development Kit Contents . . . . .	1
1.2 Requirements . . . . .	1
1.3 Additional Documents . . . . .	1
1.4 Part Identification . . . . .	1
2 Quick Start . . . . .	2
2.1 Set Up System . . . . .	2
2.2 Copy the Example Projects . . . . .	2
2.3 Running an Example Project . . . . .	2
3 Software . . . . .	3
3.1 CyanIDE . . . . .	3
3.2 USB eICE Drivers . . . . .	3
4 Installing CyanIDE . . . . .	4
4.1 From the Cyan Tools CD-ROM . . . . .	4
4.2 From the Cyan Website . . . . .	4
4.3 Installing the USB eICE Driver . . . . .	4
5 CyanIDE Examples . . . . .	5
5.1 Development Board Examples . . . . .	5
5.2 Analogue Examples . . . . .	7
5.3 Ethernet Examples . . . . .	9
5.4 EMI Examples . . . . .	10
5.5 General Examples . . . . .	11
5.6 GPIO Examples . . . . .	12
5.7 LCD Examples . . . . .	13
5.8 Serial Examples . . . . .	13
5.9 USB Examples . . . . .	14
5.10 Simulator Examples . . . . .	17
6 Development Board . . . . .	18
6.1 Overview . . . . .	18
6.2 Description . . . . .	19
7 USB eICE Adaptor . . . . .	20
7.1 Overview . . . . .	20
7.2 Cyan Technology USB eICE Adaptor . . . . .	20
7.3 CyanTools USB eICE Adaptor . . . . .	21

8	Connections and Pin Headers . . . . .	22
8.1	J18: Power Supply. . . . .	22
8.2	P3: eICE Debug Port. . . . .	22
8.3	Serial Ports . . . . .	23
8.4	Analogue Inputs and Outputs . . . . .	24
8.5	Memory Configuration . . . . .	26
8.6	I2S Audio Codec . . . . .	28
8.7	P22: Motor Control PWM. . . . .	29
8.8	Other Headers. . . . .	30
	Appendix A Important Notes . . . . .	35
	Appendix B Circuit Diagrams . . . . .	36
	B.1 Development Board. . . . .	36
	B.2 Daughter Board (eCOG1X14Z5). . . . .	46
	Appendix C Board Layouts. . . . .	50
	C.1 Development Board. . . . .	50
	C.2 Daughter Board (eCOG1X14Z5). . . . .	51

# List of Tables

1:	J18: Power supply configuration . . . . .	22
2:	P3: eICE connections . . . . .	22
3:	P1: Serial port 1A . . . . .	23
4:	P2: Serial port 2B . . . . .	23
5:	S1: Serial port 1B . . . . .	23
6:	S2: Serial port 2A . . . . .	23
7:	S2A, S2B, S3A, S3B: Analogue inputs . . . . .	24
8:	J3: Analogue signals . . . . .	24
9:	J5: ADC1_Vin1 input selection . . . . .	25
10:	J6: ADC2_Vin1 input selection . . . . .	25
11:	J4, J14: Flash memory data bus width . . . . .	26
12:	J7: CS1 address map . . . . .	26
13:	J10: CS0 memory selection. . . . .	26
14:	J15: Memory expansion . . . . .	26
15:	J8: Codec clock source . . . . .	28
16:	J16: Serial port isolation . . . . .	28
17:	S8: Line input and output. . . . .	28
18:	S4: Microphone input. . . . .	29
19:	S9: Headphone output. . . . .	29
20:	P22: MCPWM connections . . . . .	29
21:	J11: Camera module header . . . . .	30
22:	J12: SIM card . . . . .	30
23:	J13: Smart card . . . . .	31
24:	J16: Serial port isolation . . . . .	31
25:	J29: Radio module header . . . . .	32
26:	J32: External ULPI USB signals . . . . .	32
27:	J33: Internal USB PHY signals . . . . .	32
28:	J2: Digital I/O . . . . .	33
29:	J27: Digital I/O . . . . .	33
30:	J25: LEDs 1-4 . . . . .	34
31:	J26: LEDs 5-8 . . . . .	34

# 1 Introduction

## 1.1 Development Kit Contents

- eCOG1X development board
- eCOG1X14Z5 daughter board
- USB eICE adaptor
- USB cable (A to mini-B)
- 10 way ribbon cable
- Null modem serial cable
- CD-ROM containing CyanIDE development software and documentation
- Universal input 9V d.c. power supply with country-specific mains cable

## 1.2 Requirements

- A Windows-based PC system.  
(minimum 1GHz CPU speed, higher speed recommended).
- Windows 2000 or Windows XP operating system.
- 100MB free disk space.
- 512MB memory (1GB recommended).
- A spare USB port.
- A spare serial port (optional).
- System administrator privileges are required for software installation.

## 1.3 Additional Documents

1. eCOG1X User Manual
2. CyanIDE User Manual
3. eCOG1X Development Board User Manual
4. eCOG1 USB eICE Programming Adaptor User Manual
5. TN006 Configuring the FTDI FT2232

## 1.4 Part Identification

In this document, any reference to eCOG1 means the generic chip and is applicable to all versions. All eCOG1 devices are suffixed according to their version; any reference to a particular version such as eCOG1X is specific to that version.

## 2 Quick Start

Please also read *Appendix A Important Notes*.

### 2.1 Set Up System

- Unpack and check contents of kit.
- Install CyanIDE development software and USB device drivers.  
See section 4 *Installing CyanIDE* for more details.
- Check setting of jumper J18 on the development board is correct.  
Link J18 pins 1-2 to power the board from the external 9V d.c. power supply.  
See section 8.1 *J18: Power Supply* for other options.
- Connect the 9V d.c. power supply to the input jack S7.
- Connect the 10-way ribbon cable between the USB eICE adaptor and connector P3 on the development board.
- Connect the USB cable between the eICE adaptor and the host PC.
- Connect or turn on the mains input to the power supply.

### 2.2 Copy the Example Projects

CyanIDE includes a range of example projects for the development kits, copied during installation into the *<examples>* directory under the CyanIDE install directory, usually *<C:\Program Files\Cyan Technology\CyanIDE>*. The installation process also creates a *<CyanIDE Projects>* directory in the user's *My Documents* folder. It is recommended that the examples for the eCOG1X development kit are copied into this projects directory before use, to avoid making any changes to the original examples.

- Open the *My Documents* directory and browse into the *CyanIDE Projects* directory. This includes a shortcut to the CyanIDE examples directory.
- Open the *<CyanIDE Examples>* shortcut and then the *<eCOG1X dev board>* directory.
- Type **ctrl-A** or click **Edit->Select All** to select all the eCOG1X example projects, then type **ctrl-C** or click **Edit->Copy** to copy them to the clipboard.
- Click the **Back** button twice to return to the CyanIDE Projects directory.
- Type **ctrl-V** or click **Edit->Paste** to paste the example projects from the clipboard into the current directory.

### 2.3 Running an Example Project

As an introduction, try the LCD example project located in the LCD\LCDTest directory.

- Start CyanIDE.
- From the main menu, select **Project->Open** and browse to the LCD example directory *<CyanIDE Projects\LCD\LCDTest>*. Select the project file *<\*.cyp>* and click **Open**. CyanIDE loads the project and displays the files included in the project in the navigator pane at the left of the main window.
- Select **Build->Rebuild All** from the main menu. This compiles the project source files and links the object code into a download image file.
- Select **Debug->Run**. CyanIDE connects to the eCOG1X target processor on the development board, downloads the application code and begins execution.



## **3 Software**

### **3.1 CyanIDE**

The CyanIDE software development package supports the eCOG1 family of microcontrollers, providing project management, source code editor, C compiler, assembler, linker, source level debugger, and online help files. It is available on CD-ROM, or as a download to registered users on the Cyan web site at [www.cyantechology.com](http://www.cyantechology.com).

CyanIDE communicates with the eCOG1X microcontroller on the target system via the eICE debug port. The host PC requires a simple eICE adaptor that connects to the 10-way header P3 on the development board. The development kit includes the Cyan USB eICE adaptor.

CyanIDE includes a range of example applications for the development board. These can provide a good starting point for customer applications, or just as examples showing how to set up a software project. Further application examples including CyanIDE project files with source code are available on the web site support pages.

### **3.2 USB eICE Drivers**

The software drivers for the USB eICE adaptor are included in the CyanIDE development package. CyanIDE V1.4 or later includes the required version of the driver files as standard.

To ensure that the driver files are present, install the CyanIDE software and any necessary updates before connecting the eICE adaptor to the host PC. Further details about the software installation are shown later in this document.

Note that any previous version of CyanIDE should be uninstalled before the latest version is installed. This includes any USB device drivers for Cyan products such as the evaluation board, which should be removed via the Device Manager. Installing the new version of CyanIDE also reinstalls the USB eICE device driver.

## 4 Installing CyanIDE

### 4.1 From the Cyan Tools CD-ROM

Insert the CD into the CD-ROM drive. The html start page should load automatically in the default browser.

If the start page does not load automatically, or Autorun is disabled for the CD-ROM drive, then Browse the Cyan CD in a file manager or explorer window, and open the file `<index.htm>`.

When the start page is displayed, click on the **Menu** button, then select the **Install CyanIDE** item to begin installation of the development software. Follow the instructions presented by the installation program.

### 4.2 From the Cyan Website

- Navigate to the software downloads page on the website, located at <http://www.cyantechnology.com/support/updates.php>.
- Download the CyanIDE full version installation file to a temporary directory. Please note that users must log in to the website with their registered account name and password for the support forum to download this file.
- Execute the downloaded file to install the development software. Follow the instructions presented by the installation program.

### 4.3 Installing the USB eICE Driver

CyanIDE V1.4 includes the driver for the USB eICE adaptor as standard, and it is not necessary to install them separately.

When an eICE adaptor is connected to a PC for the first time, the USB enumeration process identifies it as new hardware, and Windows starts the *Found New Hardware* process. Driver installation under Windows 2000 and Windows XP is fully automatic and does not require any user interaction.

## 5 CyanIDE Examples

CyanIDE includes a number of example software projects, pre-configured for use with the eCOG1X development board, the eCOG1k development and evaluation boards, or the eCOG1 simulator. The examples for the eCOG1X development board are described briefly in the table below.

### 5.1 Development Board Examples

The following example applications for this version of the eCOG1X development board are included as standard with CyanIDE V1.4.

Example	Comments
<b>Analogue</b>	<b>Examples showing the use of the analogue peripheral</b>
BufferedSineOutput	Sine wave output from DAC1 using buffered routines
CounterTrigger	Using the CNT1 counter to trigger ADC conversions
DualOutput	Sending sine and triangular waves simultaneously from the two DAC outputs
SineOutput	Simple sine wave output from DAC1
SoftwareTrigger	Triggering ADC conversions in software
SoftwareTriggerSynced	Triggering two ADC conversions simultaneously in software
<b>EMAC</b>	<b>Ethernet example</b>
PacketLoopback	Send and receive an Ethernet packet using a loopback cable
<b>EMI</b>	<b>External Memory Interface examples</b>
SDRAM	Read and write data to the external SDRAM
SRAM	Read and write data to the external SRAM
<b>General</b>	<b>General purpose examples</b>
IROM1X	Downloads code to internal flash memory
Timers	Shows the use of all the eCOG1X timers
<b>GPIO</b>	<b>Examples using the GPIO connections</b>
FlashLEDs	Flashes LEDs in sequence on the development board
Interrupts	Trigger interrupts on the GPIO inputs from the DIP switches
ReadSwitches	Read the status of the DIP switches
<b>LCD</b>	<b>LCD example</b>
LCDTest	Displays all possible character combinations on the LCD
<b>Serial</b>	<b>Serial port example</b>
DuartEcho	Echoes received characters on UART1A

Example	Comments
<b>USB</b>	<b>USB examples</b>
AudioPeripheral	Sends and receives real-time streaming audio data
KeyboardHost	Receives characters from a connected USB keyboard
KeyboardPeripheral	Sends characters as a USB keyboard
MSDHost	Performs block reads and writes on a connected USB mass storage device
MSDPeripheral	Emulates a large read-only mass storage device

The examples may include more than one version of each project, with configurations for different target eCOG1X daughter boards. In the following summary descriptions of each project, the label [device] in the project file and the configuration file is replaced with the name of the target eCOG1X device, usually *eCOG1X14Z*.

### 5.1.1 Serial Port Configuration

Many examples use one of the serial ports to report results or display messages. Use a terminal program such as 'HyperTerminal' to communicate with the application. The default serial port configuration is shown below.

- 9600 Baud
- 8 data bits
- no parity bits
- one stop bit
- no flow control

## 5.2 Analogue Examples

### 5.2.1 Buffered Sine Output

This application outputs a 1kHz waveform from the DAC1 output. It uses buffered output routines so that calls to printf (which are blocking) do not interrupt the DAC output waveform. It uses UART1A to output text to a terminal.

File	Comments
BufferedSineOutput-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

### 5.2.2 Counter Trigger

This example uses CNT1 to trigger ADC conversions. This is performed in the background while the application waits for the user to press a key (a blocking operation). It uses UART1A to output text to a terminal and receive keyboard input.

File	Comments
CounterTrigger-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

### 5.2.3 Dual Output

This application outputs a 1kHz sine waveform from the DAC1 output and a 1kHz triangle waveform from the DAC2 output. The two DAC outputs are updated at the same time. It uses UART1A to output text to a terminal.

File	Comments
DualOutput-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

### 5.2.4 Sine Output

This application outputs a 1kHz sine waveform from the DAC1 output. It uses UART1A to output text to a terminal.

File	Comments
SineOutput-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

### 5.2.5 Software Trigger

This example performs ADC conversions by triggering the conversion from software. It uses UART1A to output text to a terminal.

File	Comments
SoftwareTrigger-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

### 5.2.6 Software Trigger Synced

This example performs ADC conversions by triggering the conversion from software. ADC2 is synchronised to ADC1 so that both conversions take place at the same time. It uses UART1A to output text to a terminal.

File	Comments
SoftwareTriggerSynced-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

## 5.3 Ethernet Examples

### 5.3.1 Packet Loopback

This application performs simple packet transmit and receive using the Ethernet MAC. The contents of the received packet are printed to the serial port output. In order to loop back the packets, connect a suitable Ethernet loopback cable to the RJ45 connector S3. The example uses UART1A to output text to a terminal.

File	Comments
PacketLoopback-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

## 5.4 EMI Examples

### 5.4.1 SDRAM

An SDRAM test for the eCOG1X development board, performing data writes to and reads from the external SDRAM device connected to the EMI bus. It uses UART1A to output text to a terminal.

File	Comments
SDRAM-eCOG1X14Z.cyp	CyanIDE project file
cstartup.asm	C environment initialisation
eCOG1X14Z.cfg	Peripheral configuration file
irom.map	Memory map
main.c	Application code
putchar.c	Implementation of <code>putchar()</code> for output to UART1A
putchar.h	Function prototypes for UART1A access routines
sdraminit.asm	SDRAM custom initialisation code

### 5.4.2 SRAM

An SRAM test for the eCOG1X development board, performing data writes to and reads from the two external SRAM devices connected to the EMI bus. It uses UART1A to output text to a terminal.

File	Comments
SRAM-eCOG1X14Z.cyp	CyanIDE project file
cstartup.asm	C environment initialisation
eCOG1X14Z.cfg	Peripheral configuration file
irom.map	Memory map
main.c	Application code
putchar.c	Implementation of <code>putchar()</code> for output to UART1A
putchar.h	Function prototypes for UART1A access routines



## 5.5 General Examples

### 5.5.1 IROM1X

Code download and programming utility for the eCOG1X internal flash memory.

File	Comments
IROM1X.cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
iram.map	Memory map
flash.c	Application code
FlashProg.c	Flash programming routines
FlashProg.h	Function prototypes for flash programming routines
initfunc.py	MMU initialisation function to map internal RAM as code
irq.asm	Interrupt vector table
StdDefs.h	Common symbol definitions

### 5.5.2 Timers

This example demonstrates the use of the timers available in the eCOG1X. It uses UART1A to output text to a terminal.

File	Comments
Timers-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

## 5.6 GPIO Examples

### 5.6.1 Flash LEDs

This application flashes the LEDs on the development board in sequence. It requires an appropriate interconnection daughter board fitted to headers J2 and J27 to connect the LEDs to the eCOG1X GPIO signals. It uses UART1A to output text to a terminal.

File	Comments
FlashLEDs-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

### 5.6.2 Interrupts

This example demonstrates interrupts on the GPIO inputs. It uses the DIP switches on the development board to trigger the interrupts. It requires an appropriate interconnection daughter board fitted to headers J2 and J27 to connect the switches to the eCOG1X GPIO signals. It uses UART1A to output text to a terminal.

File	Comments
Interrupts-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

### 5.6.3 Read Swithes

This example demonstrates reading the states of the GPIO inputs. It uses the DIP switches on the development board to set the input states. It requires an appropriate interconnection daughter board fitted to headers J2 and J27 to connect the switches to the eCOG1X GPIO signals. It uses UART1A to output text to a terminal.

File	Comments
ReadSwitches-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

## 5.7 LCD Examples

### 5.7.1 LCD Test

Initially this application performs a diagnostic test on the LCD, showing all possible display segment combinations on each digit. Once this test is complete, it displays a time value in minutes and seconds (mm:ss) using the TMR timer peripheral to generate 1s tick interrupts. It uses UART1A to output text to a terminal.

File	Comments
LCDTest-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

## 5.8 Serial Examples

### 5.8.1 DUART Echo

This example echoes back out any incoming characters from the UART1A serial port. It uses the *serial\_lib* routines to provide circular buffering and interrupt support.

File	Comments
DuartEcho-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code

## 5.9 USB Examples

### 5.9.1 Audio Peripheral

The audio peripheral example shows the eCOG1X working as a pseudo-soundcard with line-in and line-out connections. When the host requests audio from line-in, the eCOG1X provides a 1kHz sine wave sampled at 44.1kHz, 16 bit stereo. Audio sent from the host is displayed via the serial port as a simple text VU meter, showing the audio level.

This example demonstrates the following:

- Using the peripheral audio library plugin to implement most of the required functionality.
- Using the default control endpoint (EP0).
- Using the isochronous endpoint (EP2) for both data input and output.
- Using DMA or FIFO buffering to perform data transfers.
- Returning text strings on request.

The example uses the peripheral audio plugin to perform most of the USB event handling.

File	Comments
AudioPeripheral-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code
USB\usb_plugin_audio_peripheral_callbacks.c	Configuration code

### 5.9.2 Keyboard Host

The keyboard host example shows the eCOG1X working as a USB host which accepts keystrokes from most USB keyboards.

This example demonstrates the following:

- Using the host keyboard library plugin to implement most of the required functionality.
- Using the default control endpoint (EP0).
- Using the interrupt endpoint (EP1) for data input.
- Using DMA to perform data transfers.
- Detecting a suitable peripheral keyboard with boot protocol support.

The example uses the host keyboard plugin to perform most of the USB event handling. The connected keyboard must support the boot protocol, as the host keyboard does not support parsing of arbitrary HID reports.

File	Comments
KeyboardHost-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code
USB\usb_plugin_keyboard_host_callbacks.c	Configuration code

### 5.9.3 Keyboard Peripheral

The keyboard peripheral example shows the eCOG1X taking ASCII characters from the serial port and simulating keystrokes from a keyboard to a USB host.

This example demonstrates the following:

- Using the peripheral keyboard library plugin to implement most of the required functionality.
- Using the default control endpoint (EP0).
- Using the interrupt endpoint (EP1) for data output.
- Using DMA to perform data transfers.
- Generating a user event (`USB_EVENTTYPE_USER`).
- Converting ASCII character codes to USB keystrokes.

The example uses the peripheral keyboard plugin to perform most of the USB event handling.

File	Comments
KeyboardPeripheral-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code
USB\usb_plugin_keyboard_peripheral_callbacks.c	Configuration code

### 5.9.4 MSD Host

The host mass storage example shows the eCOG1X working as a USB host which performs block reads from and writes to a connected USB flash drive.

This example demonstrates the following:

- Using the host mass storage device library plugin to implement most of the required functionality.
- Using the default control endpoint (EP0).
- Using the bulk endpoint (EP3) for both data input and output.
- Mapping different host endpoint numbers to connected peripheral endpoints.
- Detecting a suitable USB flash drive peripheral.
- Sending and receiving SCSI commands over USB.

The example uses the MSD host plugin to perform most of the USB event handling. It performs a sample block read from and write to block 1 when a suitable USB flash drive is connected.

File	Comments
KeyboardHost-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code
USB\usb_plugin_msd_host_callbacks.c	Configuration code

### 5.9.5 MSD Peripheral

The mass storage peripheral example shows the eCOG1X working as a large (256MB) read-only mass storage device. This can be used as a method for transferring large amounts of generated or captured data from the eCOG1X.

This example demonstrates the following:

- Using the peripheral mass storage device library plugin to implement most of the required functionality.
- Using the default control endpoint (EP0).
- Using bulk endpoints (EP1, EP2) for data input and output.
- Using DMA or FIFO buffering to perform data transfers.
- Returning text strings on request.
- Implementing a read-only FAT32 file system.

The example uses the MSD peripheral plugin to perform most of the USB event handling.

File	Comments
MSDPeripheral-[device].cyp	CyanIDE project file
cstartup.asm	C environment initialisation
[device].cfg	Peripheral configuration file
internal.map	Memory map
main.c	Application code
USB\usb_plugin_msd_peripheral_callbacks.c	Configuration code

## 5.10 Simulator Examples

The following example simulator applications are also included with CyanIDE. They can be found in the `<examples\simulator>` directory located below the CyanIDE installation directory.

Example	Comments
primes	Searches for prime numbers
oscillator	Example of using a simulated memory-mapped device

### 5.10.1 primes

The 'primes' application searches for prime numbers and displays results in the Debug Output Window. The results are displayed using an implementation of `putchar()` that includes the assembler PRINT instruction. The simulator interprets this instruction according to the Print mode option in Project Properties: printing a character, a hexadecimal value, or raising an exception.

File	Comments
primes.cyp	CyanIDE project file
cstartup.asm	C environment initialisation
irq.asm	Entry point and interrupt vectors
primes.c	Application code
putchar.c	Character output routine
simulator.map	Memory map

### 5.10.2 oscillator

The 'oscillator' application reads a simulated ADC which is generating data representing a sine wave, and writes a filtered value back to a memory location. The filtered value is logged to a file.

File	Comments
oscillator.cyp	CyanIDE project file
cstartup.asm	C environment initialisation
irq.asm	Entry point and interrupt vectors
oscillator.c	Application code
putchar.c	Character output routine
oscillator.py	Simulated ADC written in Python. The <b>Run on project load</b> option for this file is set.
simulator.map	Memory map - includes CUSTOM entry

## 6 Development Board

### 6.1 Overview

The eCOG1X Development Board has the following major features.

- Processor daughter board, normally fitted with the eCOG1X14Z5 microcontroller.
- 512K x 16 bit flash memory.
- 8M x 16 bit SDRAM.
- 512K x 16 SRAM.
- 10/100 Mbits/s Ethernet using the Teridian 78Q2120C PHY.
- Mini-AB connector for the on-chip USB.
- High-speed USB (480Mbits/s) using the SMSC USB3300 ULPI PHY.
- 10 way boxed header for eICE debug port (P3).
- Four RS-232 serial ports.
- Four 12-bit analogue inputs.
- Two 12-bit analogue outputs.
- I<sup>2</sup>S audio codec (WM8731S) with line in/out, microphone input and headphone output.
- Four digit seven-segment multiplexed LCD.
- I<sup>2</sup>C serial EEPROM (24LC32A).
- SIM card socket connected to smart card interface.
- Infra-red transceiver.
- Eight user/status LEDs.
- Eight user DIP switches.
- Piezo sounder.
- Various pin headers for input and output ports and additional daughter boards.
- Powered from an eICE adaptor or from a 9V d.c. input jack.



## 6.2 Description

The eCOG1X microcontroller has a 16-bit CPU architecture and a wide range of on-chip peripherals. It operates at clock speeds of up to 70MHz internally from an 8MHz crystal or 32.768kHz watch crystal. Refer to the eCOG1X User Manual for further details.

A Spansion (AMD/Fujitsu) S29JL064 device provides 1Mbyte (512Kwords) of external flash memory. This is connected to the external memory interface (EMI) in standard bus mode and is enabled by chip select CS1 for physical addresses below 0x800000 (A23 = 0).

The development board may be configured to use either SDRAM or SRAM as external memory. A Samsung K4S281632 device provides 16Mbytes (8Mwords) of SDRAM. It is connected to the EMI peripheral in SDRAM mode, with no glue logic required, and is enabled by chip select CS0. Jumper links allow SRAM to be enabled instead of SDRAM if required for low power applications. Two BS62LV4006 devices provide 1Mbyte of SRAM, organised as 512Kx16 bits.

The on-chip Ethernet MAC connects to the Teridian 78Q2120C device which supports operation at both 10Mbits/s and 100Mbits/s.

The board is fitted with a 10-way boxed header (P3) for the eICE debug port. This port connects directly to the processor core, and is used for downloading and debugging applications code. A Cyan USB eICE adaptor is supplied with the Development Board, although any external eICE adaptor may be used.

Four RS-232 serial ports are available on the Development Board. Positions P1 and P2 are fitted with 9-way D-type plugs and are wired as DTE ports, while S1 and S2 are fitted with 9-way sockets and are wired as DCE ports. A null-modem serial cable is provided to connect a PC serial port to P1 or P2. To connect a PC serial port to S1 or S2, use a standard straight-through serial cable (not supplied).

A 24LC32A I<sup>2</sup>C serial EEPROM provides a small amount of additional non-volatile storage for use in applications. It is connected to one of the two DUSART serial channels. The I<sup>2</sup>C protocol is supported by the DUSART peripheral.

Four 12-bit analogue inputs on the eCOG1X device are available on SMA connectors S2A, S2B, S3A and S3B, and are buffered with high-speed amplifiers. Eight analogue inputs are also available on pin header J3. Two further analogue inputs can be connected to onboard potentiometers VR1 and VR2 for testing or to signals from pin headers for other daughter boards, by fitting jumper links to pin headers J5 and J6. Two buffered analogue outputs are also available on pin header J3.

The Development Board can be powered from an external d.c. power supply, connected via a standard input jack (S7), or from the eICE debug connection on P3. Jumper J18 selects the power supply source.

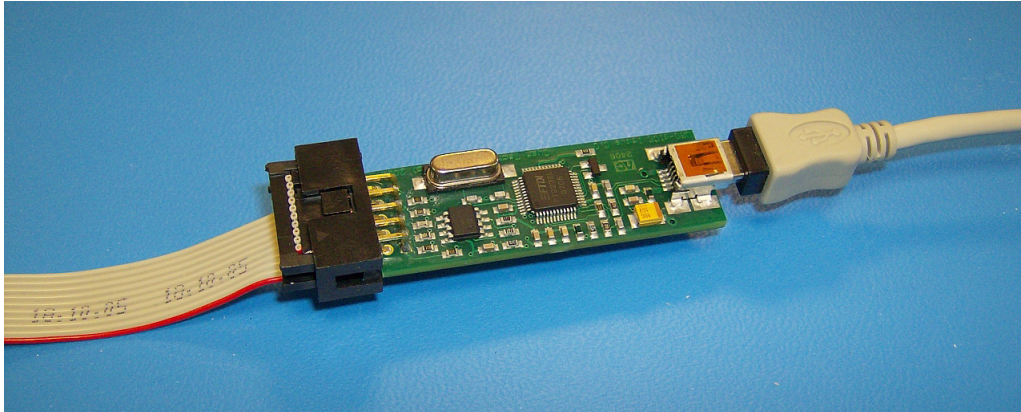
Note that when a USB eICE adaptor is used to power the board, it is unlikely that it can provide enough supply current to power all peripheral interfaces at maximum clock speeds. The maximum supply current available to a USB bus-powered slave device is only 500mA at 5V. If many high-speed peripheral functions are required, then it is recommended that the board is powered from an external d.c. power supply, nominally 9V, via the input jack.

## 7 USB eICE Adaptor

### 7.1 Overview

A USB eICE adaptor provides a connection from the host PC to the target eCOG1X device. A number of different eICE adaptors are available from Cyan Technology and from some third-party suppliers. All use the same device driver software.

### 7.2 Cyan Technology USB eICE Adaptor



**Figure 1: Cyan USB eICE adaptor**

The Cyan USB eICE adaptor has the following major features.

- FTDI FT2232 device providing USB interface.
- USB mini-B socket.
- USB cable (A to mini-B) for host PC connection.
- 10 way boxed header and ribbon cable for eICE debug port.
- Powered from USB +5V supply.
- 5V supply connection available for low-power target systems.

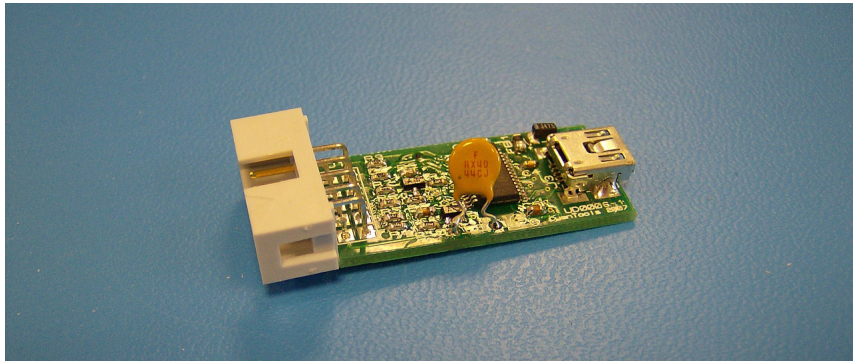
This adaptor uses the FTDI FT2232 device to implement the USB interface. A standard USB cable connects to the host side of the unit via a mini-B USB socket. The PC end of the cable has a type A USB connector fitted. The eICE connection to the eCOG1X target device is via a 10 way boxed header and ribbon cable. The FT2232 device provides the USB interface to the host PC and digital inputs and outputs for the eICE signals.

A 93C56 serial eeprom provides non-volatile storage for the FT2232 configuration data including serial numbers, USB PID and VID numbers, and identifier strings. This configuration data is required to allow the unit to identify itself to the host PC during the USB device enumeration process, and for the PC then to select the correct device driver files. The Cyan USB eICE adaptors are loaded with the required configuration data during functional test.

The adaptor is powered from the +5V supply available on the host USB connection. The FT2232 device includes an on-chip regulator which provides 3.3V for the I/O connections to the eCOG1X target device. An external transistor connects to the bidirectional signal eICE\_LOADB, used for handshaking eICE messages.

All signals to the eICE connection include 100 $\Omega$  series resistors. The LOADB signal also includes a 4.7k $\Omega$  pull-up resistor. The USB +5V supply is also connected to the 10 way header; this may be used to power small target systems provided the total current drawn is within the 500mA limit available from a standard USB host.

## 7.3 CyanTools USB eICE Adaptor



**Figure 2: CyanTools USB eICE adaptor**

The CyanTools USB eICE adaptor has the following major features.

- FTDI FT245R device providing USB interface.
- USB mini-B socket.
- USB cable (A to mini-B) for host PC connection.
- 10 way boxed header and ribbon cable for eICE debug port.
- Powered from USB +5V supply.
- 5V supply connection available for low-power target systems.

An earlier version of this adaptor has a slider switch which is used to set the target power supply connection to either 5V or 3.3V. This should be set to the 5V position for use with the eCOG1X development board.

This adaptor uses the FTDI FT245R device to implement the USB interface. A standard USB cable connects to the host side of the unit via a mini-B USB socket. The PC end of the cable has a type A USB connector fitted. The eICE connection to the eCOG1X target device is via a 10 way boxed header and ribbon cable. The FT245R device provides the USB interface to the host PC and digital inputs and outputs for the eICE signals.

The FT245R includes a serial eeprom for non-volatile storage of configuration data including serial numbers, USB PID and VID numbers, and identifier strings. This configuration data is required to allow the unit to identify itself to the host PC during the USB device enumeration process, and for the PC then to select the correct device driver files.

The adaptor is powered from the +5V supply available on the host USB connection. The FT245R device includes an on-chip regulator which provides 3.3V for the I/O connections to the eCOG1X target device. An external transistor connects to the bidirectional signal eICE\_LOADB, used for handshaking eICE messages.

All signals to the eICE connection include 100 $\Omega$  series resistors. The LOADB signal also includes a 4.7k $\Omega$  pull-up resistor. The USB +5V supply is also connected to the 10 way header; this may be used to power small target systems provided the total current drawn is within the 500mA limit available from a standard USB host.

## 8 Connections and Pin Headers

### 8.1 J18: Power Supply

The eCOG1X development board is normally powered from an external d.c. supply via the input jack S7. It can also be powered from the eICE adaptor. Header J18 selects which power supply source is used to power the board.

The external power supply from the input jack S7, nominally 9V, is regulated down to 5V initially. This 5V supply is connected to J18 pin 1. To power the board from the regulated external supply, link J18 pins 1-2. This connects the regulated input 5V supply to a second 3.3V regulator which powers the rest of the board. The processor daughter board includes a second local regulator which generates the 1.8V supply voltage for the processor core and analogue sections from the 3.3V supply.

The power connection from the eICE adaptor is connected to J18 pin 3. This can be used to power the board from the supply provided by the eICE adaptor, instead of from an external supply. Check the voltage of the power supply output on the eICE adaptor. If it is 5V, then link J1 pins 2-3 to connect this 5V supply to the input of the 3.3V regulator. If it is 3.3V, then link J1 pins 3-4 to bypass the 3.3V regulator and connect directly to the development board 3.3V power supply rail.

The Cyan USB eICE adaptor provides 5V d.c. on the VDD\_EICE pin, from the USB hub on the host PC. To use this to power the board, connect J18 pins 2-3.

Note that it is likely that an external supply will be necessary when using several high-performance peripheral functions or external devices, as they may require more supply current than is available from a normal USB connection.

Mode	Link
External 9V PSU (normal operation)	J18 pins 1-2
Powered from Cyan USB eICE adaptor (5V output)	J18 pins 2-3
Other eICE adaptor (3.3V output)	J18 pins 3-4
External PSU and supply 3.3V to eICE adaptor	J18 pins 1-2, 3-4

**Table 1: J18: Power supply configuration**

### 8.2 P3: eICE Debug Port

The pin connections for the eICE signals on the 10 way boxed header P3 are shown in the table below.

Pin	Name	Description
1	eICE_MOSI	Master Out Slave In, input from the eICE adaptor.
2	VDD_EICE	Either 5V or 3.3V input from the eICE adaptor, or 3.3V output to the eICE adaptor. See J18 for power supply configuration options.
3	NC	No connection.
4	GND	
5	eICE_LOADB	eICE LoadB signal, open-collector, bidirectional.
6	GND	
7	eICE_CLOCK	eICE serial clock, input from the eICE adaptor.
8	GND	
9	eICE_MISO	Master In Slave Out, output to the eICE adaptor.
10	eICE_nRESET	Active low reset, connected in parallel with the reset switch.

**Table 2: P3: eICE connections**

The Cyan USB eICE adaptor provides 5V d.c. on the VDD\_EICE pin, from the USB hub on the host PC. To use this to power the board, connect J18 pins 2-3.

## 8.3 Serial Ports

P1, P2, S1 and S2 are standard PC compatible 9-way D-type serial connectors. P1 and P2 are plugs, wired as DTE, and require a null-modem cable to connect to a PC. S1 and S2 are sockets, wired as a DCE, and require a standard straight-through cable to connect to a PC. S1 and P2 also provide support for RTS/CTS hardware handshake signals using two GPIO lines each.

The normal eCOG1X14Z5 daughter board has UART1A connected to P1, UART1B to S1, UART2A to S2 and UART2B to P2.

### 8.3.1 P1: Serial Port 1A

Pin	Name	Description	Port	Function
2	RX1A	Receive data input	PortS_1	UART1A_RX
3	TX1A	Transmit data output	PortS_0	UART1A_TX
5	GND			
1, 4, 6-9	NC	No connection		

**Table 3: P1: Serial port 1A**

### 8.3.2 P2: Serial Port 2B

Pin	Name	Description	Port	Function
2	RX2B	Receive data input	PortS_7	UART2B_RX
3	TX2B	Transmit data output	PortS_6	UART2B_TX
5	GND			
7	RTS2B	Ready-to-send output.	PortS_5	GPIO5_5
8	CTS2B	Clear-to-send input	PortS_4	GPIO5_4
1, 4, 6, 9	NC	No connection		

**Table 4: P2: Serial port 2B**

### 8.3.3 S1: Serial Port 1B

Pin	Name	Description	Port	Function
2	TX1B	Transmit data output	PortS_2	UART1B_TX
3	RX1B	Receive data input	PortS_3	UART1B_RX
5	GND			
7	CTS1B	Clear-to-send input	PortS_0	GPIO5_0
8	RTS1B	Ready-to-send output.	PortS_1	GPIO5_1
1, 4, 6, 9	NC	No connection		

**Table 5: S1: Serial port 1B**

### 8.3.4 S2: Serial Port 2A

Pin	Name	Description	Port	Function
2	TX2A	Transmit data output	PortS_6	UART2A_TX
3	RX2A	Receive data input	PortS_7	UART2A_RX
5	GND			
1, 4, 6-9	NC	No connection		

**Table 6: S2: Serial port 2A**

## 8.4 Analogue Inputs and Outputs

Notes:

- The analogue ground AGND and the digital ground GND are not connected together on the development kit main board, only on the processor daughter board near the analogue inputs.
- Analogue input channels ADC1\_Vin1 and ADC2\_Vin1 are connected to onboard signals or test potentiometers, configured by pin headers J5 and J6.

### 8.4.1 S2A, S2B, S3A, S3B: Analogue Inputs

Coaxial connectors S2A, S2B, S3A and S3B provide four analogue input channels to the on-chip 12-bit ADCs. These input signals are buffered by high-speed op-amps IC6 and IC19.

Pin	Name	Description	Port
S2A	VIN2A	ADC1 input channel 2	ADC1_Vin2
S2B	VIN2B	ADC2 input channel 2	ADC2_Vin2
S3A	VIN3A	ADC1 input channel 3	ADC1_Vin3
S3B	VIN3B	ADC2 input channel 3	ADC2_Vin3

**Table 7: S2A, S2B, S3A, S3B: Analogue inputs**

### 8.4.2 J3: Analogue Signals

Pin header J3 provides access to a further eight (unbuffered) input channels to the ADCs and to the two analogue output channels from the DACs, buffered by IC14. It also includes connections for power, ground, and the reference voltage Vref.

Pin	Name	Description	Port
1	VIN4A	ADC1 input channel 4	ADC1_Vin4
2	VIN5A	ADC1 input channel 5	ADC1_Vin5
3	AGND	Analogue GND	
4	AGND	Analogue GND	
5	VIN6A	ADC1 input channel 6	ADC1_Vin6
6	VIN7A	ADC1 input channel 7	ADC1_Vin7
7	AGND	Analogue GND	
8	AGND	Analogue GND	
9	VIN4B	ADC2 input channel 4	ADC2_Vin4
10	VIN5B	ADC2 input channel 5	ADC2_Vin5
11	AGND	Analogue GND	
12	AGND	Analogue GND	
13	VIN6B	ADC2 input channel 6	ADC2_Vin6
14	VIN7B	ADC2 input channel 7	ADC2_Vin7
15	AGND	Analogue GND	
16	AGND	Analogue GND	
17	VREF	Reference voltage (+1.22V)	Vref
18	+3.3V		
19	VOUTA	DAC1 analogue output	DAC1
20	VOUTB	DAC2 analogue output	DAC2

**Table 8: J3: Analogue signals**

### 8.4.3 J5: ADC1\_Vin1 Input Selection

Header J5 allows ADC1 input channel 1 to be connected either to the analogue input signal from the radio module header (J29) or to the onboard potentiometer VR1.

Pin	Name	Description	Port
1	RADIO_VIN	Analogue input from radio module header J29	
2	VIN1A	ADC1 input channel 1	ADC1_Vin1
3	VR1	Potentiometer VR1	
4	AGND	Analogue ground	

**Table 9: J5: ADC1\_Vin1 input selection**

### 8.4.4 J6: ADC2\_Vin1 Input Selection

Header J6 allows ADC2 input channel 1 to be connected either to the analogue input signal from the USB Vbus monitor point or to the onboard potentiometer VR2.

Pin	Name	Description	Port
1	VBUS_MON	USB Vbus monitor voltage	
2	VIN2A	ADC2 input channel 1	ADC2_Vin1
3	VR2	Potentiometer VR2	
4	AGND	Analogue ground	

**Table 10: J6: ADC2\_Vin1 input selection**

## 8.5 Memory Configuration

### 8.5.1 J4, J14: Flash Memory Data Bus Width

Headers J4 and J14 are used to configure the flash memory IC4 for either 8-bit or 16-bit data bus operation. Note that if the board is configured to use the external 16-bit wide SRAM, then the flash must also be configured for a 16-bit data bus width.

Flash Data Bus Width	Link
16 bit data bus	J4 pins 1-2, J14 pins 1-2
8 bit data bus	J4 open, J14 pins 2-3

**Table 11: J4, J14: Flash memory data bus width**

### 8.5.2 J7: CS1 Address Map Selection

Header J7 configures the address map arrangement for chip select CS1.

In normal operation, the address range is split between the onboard external flash memory and the EMI expansion header J15. Physical addresses between 0x000000 and 0x7FFFFFFF (A23=0) select the flash memory, and physical addresses 0x800000 and higher (A23=1) select the expansion header bus.

Alternatively, the external flash can be disabled and the whole of the address range on CS1 used for the EMI expansion header.

Mode	Link
Normal (flash + expansion header)	J7 pins 1-2 and 3-4
Flash disabled, expansion header only	J7 pins 2-3

**Table 12: J7: CS1 address map**

### 8.5.3 J10: CS0 Memory Type Selection

Header J10 allows either the SRAM or the SDRAM to be used as the external memory on chip select CS0. The two SRAM devices provide a total of 512Kx16 bits of memory, and the SDRAM device provides 8Mx16 bits.

Mode	Link
SDRAM (8Mx16)	J10 pins 1-2
SRAM (512Kx16)	J10 pins 2-3

**Table 13: J10: CS0 memory selection**

### 8.5.4 J15: Memory Expansion

Header J15 provides connections to all the EMI (external memory interface) signals from the eCOG1X device for use with any external memory-mapped hardware. It has a 16-bit data bus and a 16-bit address bus. An external address latch may be used to hold the high address bits A16-A23 if required.

Pin	Name	Description	Port	Function
1	A0	Address bit A0	PortE_0	EMI_A0
2	A1	Address bit A1	PortE_1	EMI_A1
3	A2	Address bit A2	PortE_2	EMI_A2
4	A3	Address bit A3	PortE_3	EMI_A3
5	A4	Address bit A4	PortE_4	EMI_A4
6	A5	Address bit A5	PortE_5	EMI_A5
7	A6	Address bit A6	PortE_6	EMI_A6
8	A7	Address bit A7	PortE_7	EMI_A7

**Table 14: J15: Memory expansion**



Pin	Name	Description	Port	Function
9	+3.3V			
10	+3.3V			
11	A8	Address bit A8	PortF_0	EMI_A8
12	A9	Address bit A9	PortF_1	EMI_A9
13	A10	Address bit A10	PortF_2	EMI_A10
14	A11	Address bit A11	PortF_3	EMI_A11
15	A12	Address bit A12	PortG_0	EMI_A12
16	A13	Address bit A13	PortG_1	EMI_A13
17	A14	Address bit A14	PortG_2	EMI_A14/DQML
18	A15	Address bit A15	PortG_3	EMI_A15/DQMH
19	GND			
20	GND			
21	D0	Data bit D0	PortH_0	EMI_D0
22	D1	Data bit D1	PortH_1	EMI_D1
23	D2	Data bit D2	PortH_2	EMI_D2
24	D3	Data bit D3	PortH_3	EMI_D3
25	D4	Data bit D4	PortH_4	EMI_D4
26	D5	Data bit D5	PortH_5	EMI_D5
27	D6	Data bit D6	PortH_6	EMI_D6
28	D7	Data bit D7	PortH_7	EMI_D7
29	GND			
30	GND			
31	D8	Data bit D8	PortI_0	EMI_D8/A16
32	D9	Data bit D9	PortI_1	EMI_D9/A17
33	D10	Data bit D10	PortI_2	EMI_D10/A18
34	D11	Data bit D11	PortI_3	EMI_D11/A19
35	D12	Data bit D12	PortI_4	EMI_D12/A20
36	D13	Data bit D13	PortI_5	EMI_D13/A21
37	D14	Data bit D14	PortI_6	EMI_D14/A22
38	D15	Data bit D15	PortI_7	EMI_D15/A23
39	GND			
40	GND			
41	CS0	Chip select CS0	PortD_0	EMI_CS0
42	CS_MEM_CS1	Chip select CS1 from J87	PortD_1	EMI_CS1
43	DS1/WS1/RAS	Data strobe 1 Write strobe 1 Row address strobe	PortJ_0	DS1/WS1/RAS
44	DS0/WS0/CAS	Data strobe 0 Write strobe 0 Column address strobe	PortD_3	DS0/WS0/CAS
45	RW/RS/WEN	Read/Write direction Read strobe Write enable	PortD_2	RW/RS/WEN
46	CKE	EMI clock enable	PortJ_1	EMI_CKE
47	nWAIT	Wait request	PortJ_2	EMI_WAIT
48	CLK	EMI Clock	PortJ_3	EMI_CLK
49	+5V			
50	+5V			

**Table 14: J15: Memory expansion**

## 8.6 I2S Audio Codec

### 8.6.1 J8: Codec Clock Source

Header J8 allows the codec master clock input to be generated locally by a crystal oscillator or supplied from the eCOG1X I2S\_MCLK signal.

Mode	Link
X4 crystal oscillator (12.288MHz)	J8 pins 1-2
I2S_MCLK (PortS_5)	J8 pins 2-3

**Table 15: J8: Codec clock source**

### 8.6.2 J9: Codec Interface

Header J9 is used to isolate the onboard audio codec (IC15) from the eCOG1X port pins if they are required for other functions, as there is no other way to disable the connections to the codec. J9 pins 1-14 allow signals to the codec to be connected or removed. Link J9 pins 1-2, 3-4, etc. up to pins 13-14 to connect the codec to the eCOG1X port signals. Remove these links to disconnect the onboard codec and allow the signals on this header to be used with a different codec if required. Pins 15-20 on J9 provide additional signals which do not need to be disconnected from the onboard codec but may be required for an external codec.

Function	Port	J9 Pins		Name	Description	IC15
GPIOQ_7	PortQ_7	1	2	CSB	Codec chip select	Pin 22
I2S_SCLK	PortS_0	3	4	BCLK	I2S serial bit clock	Pin 3
I2S_WS	PortS_1	5	6	ADCLRC	ADC left/right select	Pin 7
I2S_WS	PortS_1	7	8	DACLRC	DAC left/right select	Pin 5
I2S_SD_IN	PortS_3	9	10	ADCDAT	I2S serial data input	Pin 6
I2S_SD_OUT	PortS_2	11	12	DACDAT	I2S serial data output	Pin 4
I2S_ALT_CLK_IN	PortS_4	13	14	CLKOUT	Master clock	Pin 2
SPI_CLK	PortT_0	15		SCLK	Serial clock	Pin 24
SPI_MOSI	PortT_1	16		SDIN	Serial data	Pin 23
		17		+3.3V		
		18		+3.3V		
		19		GND		
		20		GND		

**Table 16: J16: Serial port isolation**

### 8.6.3 S8: Line Input and Output

The quad phono socket S8 provides connections to the audio line input and output channels on the WM8731L audio codec.

S8 Connector	Description
Top left	Line out (left channel)
Bottom left	Line out (right channel)
Top right	Line in (left channel)
Bottom right	Line in (right channel)

**Table 17: S8: Line input and output**

### 8.6.4 S4: Microphone Input

The 3.5mm jack socket S4 provides a connection to the microphone input channel on the WM8731L audio codec.

Pin	Description
S4 tip	Microphone input
S4 ring	No connection
S4 sleeve	Microphone ground

**Table 18: S4: Microphone input**

### 8.6.5 S9: Headphone Output

The 3.5mm jack socket S4 provides a connection to the headphone output channel on the WM8731L audio codec.

Pin	Description
S9 tip	Headphone output (left channel)
S9 ring	Headphone output (right channel)
S9 sleeve	Headphone ground

**Table 19: S9: Headphone output**

## 8.7 P22: Motor Control PWM

The pin connections for the motor control PWM output signals on the 10 way boxed header P22 are shown in the table below.

These signals are used elsewhere on the board; check the links on other headers and disable any other use if these signals are required here.

Pin	Name	Description	Port	Function
1	+3.3V			
2	+3.3V			
3	MCPWM1	PWM output channel 1	PortR_0	MCPWM1
4	MCPWM2	PWM output channel 2	PortR_1	MCPWM2
5	MCPWM3	PWM output channel 3	PortR_2	MCPWM3
6	MCPWM4	PWM output channel 4	PortR_3	MCPWM4
7	MCPWM5	PWM output channel 5	PortR_4	MCPWM5
8	MCPWM6	PWM output channel 6	PortR_5	MCPWM6
9	GND			
10	GND			

**Table 20: P22: MCPWM connections**

## 8.8 Other Headers

### 8.8.1 J11: Camera Interface

Header J11 provides access to a range of digital I/O signals and is intended for use with modules or daughter boards fitted with a CMOS camera or image sensor.

These signals are used elsewhere on the board; check the links on other headers and disable any other use if these signals are required here.

Pin	Name	Description	Port	Function
1	+3.3V			
2	+3.3V			
3	SIOC/SERIAL_CLK1	Serial clock Also connected to I2C EEPROM IC9 pin 6	PortK_0	GPIOK_0
4	SIOD/SERIAL_DTA1	Serial data Also connected to I2C EEPROM IC9 pin 5	PortK_1	GPIOK_1
5	CAMERA_D0	Data bit D0	PortP_0	PIOA_0
6	CAMERA_D1	Data bit D1	PortP_1	PIOA_1
7	CAMERA_D2	Data bit D2	PortP_2	PIOA_2
8	CAMERA_D3	Data bit D3	PortP_3	PIOA_3
9	CAMERA_D4	Data bit D4	PortP_4	PIOA_4
10	CAMERA_D5	Data bit D5	PortP_5	PIOA_5
11	CAMERA_D6	Data bit D6	PortP_6	PIOA_6
12	CAMERA_D7	Data bit D7	PortP_7	PIOA_7
13	CAMERA_VSYNC	Vertical SYNC	PortQ_0	GPIOQ_0
14	CAMERA_HREF	Horizontal reference	PortQ_1	GPIOQ_1
15	CAMERA_PWDN	Power down	PortQ_2	GPIOQ_2
16	CAMERA_RESET	Reset	PortQ_3	GPIOQ_3
17	CAMERA_PCLK	Data clock from camera	PortL_3	GPIO_L_3
18	CAMERA_CLK	Clock output to camera (24MHz)	PortR_6	PWM1
19	GND			
20	GND			

**Table 21: J11: Camera module header**

### 8.8.2 J12: SIM Card

Header J12 provides access to the signals for the SIM card holder, driven by one of the smart card interface peripherals.

Pin	Name	Description	Port	Function
1	SIM_PWR_EN	SIM card power enable control output	PortN_2	SCB_PWR_EN
2	SIM_DATA	SIM card serial data (bidirectional)	PortN_4	SCB_DATA
3	SIM_RESET	SIM card reset output	PortN_1	SCB_RESET
4	SIM_CLK	SIM card clock output	PortN_0	SCB_CLK
5	SIM_CARD_IN	SIM card present switch input	PortN_3	SCB_CARD_IN

**Table 22: J12: SIM card**

### 8.8.3 J13: Smart Card

Header J13 provides connections for an external smart card or SIM card, driven by the second smart card interface peripheral.

Pin	Name	Description	Port	Function
1	+3.3V			
2	GND			
3	SC_CLK	Smart card clock output	PortN_5	SCA_CLK
4	SC_DATA_IN	Smart card serial data input or I/O	PortL_0	SCA_DATA
5	SC_DATA_OUT	Smart card serial data output (not used)		
6	SC_RESET	Smart card reset output	PortN_6	SCA_RESET
7	SC_CARD_IN	Smart card present switch input	PortL_1	SCA_CARD_IN
8	SC_PWR_EN	Smart card power enable control output	PortN_7	SCA_PWR_EN
9	SEL_3V	Select 3V or 5V electrical interface	PortK_2	GPIOK_2
10	NC	No connection		

**Table 23: J13: Smart card**

### 8.8.4 J16: Serial Port Isolation

Header J16 links the eCOG1X serial port signals for UART1A and UART1B to the first serial port RS-232 transceiver device connected to P1 and S1. To connect serial port P1 to UART1A and serial port S1 to UART1B (without hardware handshake on UART1B), link J16 pins 1-2, 3-4, 7-8 and 9-10. To connect serial port S1 to UART1B with hardware handshake (and disconnect serial P1 from UART1A), link J16 pins 3-4, 5-6, 9-10 and 11-12. To disconnect the RS-232 transceiver completely and leave these port pins available for use elsewhere, remove all links from J16.

Function	Port	J16 Pins		Name	Description	Connector
UART1A_TX	PortS_0	1	2	TX1A	Transmit data output	P1 pin 3
UART1B_TX	PortS_2	3	4	TX1B	Transmit data output	S1 pin 2
GPIO1_1	PortS_1	5	6	RTS1B	Ready-to-send output.	S1 pin 8
UART1A_RX	PortS_1	7	8	RX1A	Receive data input	P1 pin 2
UART1B_RX	PortS_3	9	10	RX1B	Receive data input	S1 pin 3
GPIO1_0	PortS_0	11	12	CTS1B	Clear-to-send input	S1 pin 7

**Table 24: J16: Serial port isolation**

The second RS-232 transceiver connected to P2 and S2 is always linked to the eCOG1X serial port signals for UART2A and UART2B.

### 8.8.5 J29: Radio Module

Header J29 provides access to a mixture of signals and is intended for use with modules or daughter boards that require a range of different functions, such as modules for ISM radio or wireless datalinks. One of the analogue input channels is also available here, as some radio modules provide an analogue signal strength indicator output.

These signals are used elsewhere on the board; check the links on other headers and disable any other use if these signals are required here.

Pin	Name	Description	Port	Function
1	+3.3V			
2	GND			
3	SPI_CLK	SPI serial clock	PortT_0	SPI_SCLK
4	RX1A	Serial port receive data input	PortS_1	UART1A_RX
5	SPI_MOSI	SPI serial data output	PortT_1	SPI_MOSI
6	TX1A	Serial port transmit data output	PortS_0	UART1A_TX
7	SPI_MISO	SPI serial data input	PortT_2	SPI_MISO
8	CTS1B	Clear-to-send input	PortS_0	GPIO0_0
9	SPI_CS_RADIO	SPI chip select	PortT_3	SPI_CS0
10	RTS1B	Ready-to-send output	PortS_1	GPIO0_1
11	PWM2	PWM2 output	PortR_7	PWM2
12	RX1B	Serial port receive data input	PortS_3	UART1B_RX
13	RADIO_VIN	Received signal strength	ADC1_Vin1	ADC1_Vin1
14	TX1B	Serial port transmit data output	PortS_2	UART1B_TX
15	+3.3V			
16	GND			

**Table 25: J29: Radio module header**

### 8.8.6 J32, J33: USB Signals

Header J32 provides a convenient location to probe the high speed USB signals from the SMSC USB3300 device, and to tie the ID pin low if required. Similarly, header J33 provides a convenient location to probe the signals from the eCOG1X internal USB port.

Pin	Name	Description
1	USB_VBUS	ULPI USB Vbus
2	USB_DATA-	ULPI DM: data (inverted)
3	USB_DATA+	ULPI DP: data (true)
4	USB_ID	ULPI ID input
5	GND	

**Table 26: J32: External ULPI USB signals**

Pin	Name	Description
1	USB_VBUS	USB Vbus divided by 5.545, connected to J6 pin 1
2	USB_DATA-	DM: data (inverted)
3	USB_DATA+	DP: data (true)
4	USB_ID	ID input
5	GND	

**Table 27: J33: Internal USB PHY signals**

### 8.8.7 J2, J27: Digital Inputs and Outputs

Headers J2 and J27 provide access to a number of signals for general-purpose use. These signals may be used elsewhere on the board; check the links on other headers and disable any other use if these signals are required here.

Note that with the eCOG1X14Z5 daughterboard, most port pins are used for peripheral functions and there are very few pins spare for use as GPIO to drive the LEDs and read the DIP switches. A small daughterboard is available which interconnects pins on headers J2 and J27 to link the LEDs and switches into signals for other port pins. Without this extra daughterboard fitted, the LEDs and switches are not connected to any port pins on the eCOG1X. With this board fitted to J2 and J27, the LEDs and switches are connected to the PortR\_x pins on the eCOG1X in parallel with the PWM and MCPWM signals.

In the following tables, the signal connections for the LEDs and DIP switches when this additional interconnection board is fitted to J2 and J27 are shown in red.

Pin	Name	Description	Port	Function
1	MCPWM1	Motor control PWM output channel 1 - P22 pin 3	PortR_0	MCPWM1
2	MCPWM2	Motor control PWM output channel 2 - P22 pin 4	PortR_1	MCPWM2
3	MCPWM3	Motor control PWM output channel 3 - P22 pin 5	PortR_2	MCPWM3
4	MCPWM4	Motor control PWM output channel 4 - P22 pin 6	PortR_3	MCPWM4
5	MCPWM5	Motor control PWM output channel 5 - P22 pin 7	PortR_4	MCPWM5
6	MCPWM6	Motor control PWM output channel 6 - P22 pin 8	PortR_5	MCPWM6
7	SW2-1	DIP switch SW2-1	PortR_0	GPIOR_0
8	SW2-2	DIP switch SW2-2	PortR_1	GPIOR_1
9	SW2-3	DIP switch SW2-3	PortR_2	GPIOR_2
10	SW2-4	DIP switch SW2-4	PortR_3	GPIOR_3
11	SW3-1	DIP switch SW3-1	PortR_4	GPIOR_4
12	SW3-2	DIP switch SW3-2	PortR_5	GPIOR_5
13	SW3-3	DIP switch SW3-3	PortR_6	GPIOR_6
14	SW3-4	DIP switch SW3-4	PortR_7	GPIOR_7
15	+3.3V			
16	GND			

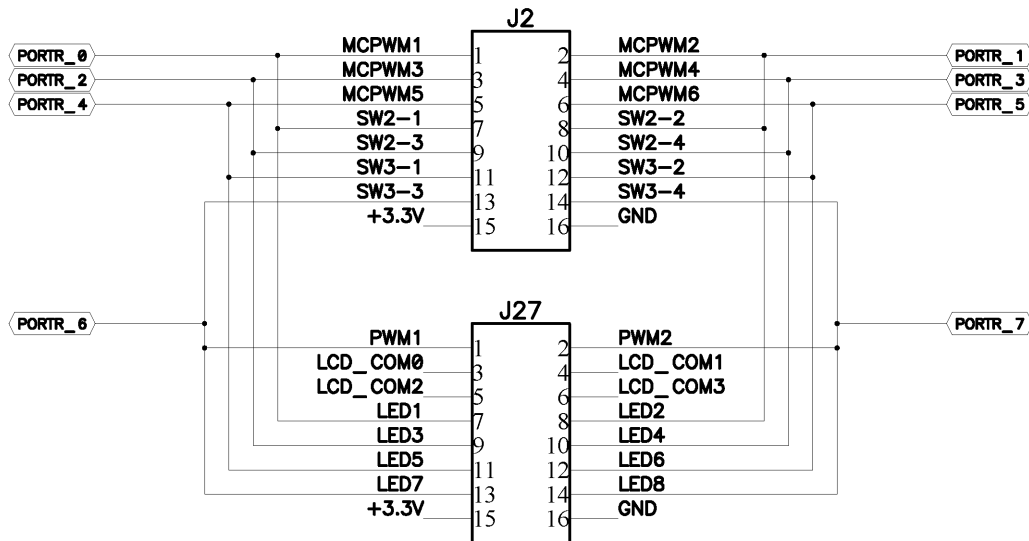
**Table 28: J2: Digital I/O**

Pin	Name	Description	Port	Function
1	PWM1	PWM1 timer output - J28 pin 1, J11 pin 18	PortR_6	PWM1
2	PWM2	PWM2 timer output - J28 pin 3, J29 pin 11	PortR_7	PWM2
3	LCD_COM0	LCD common 0 - LCD1 pin 11, J13 pin 4	PortL_0	LCD_COM0
4	LCD_COM1	LCD common 1 - LCD1 pin 20, J13 pin 7	PortL_1	LCD_COM1
5	LCD_COM2	LCD common 2 - LCD1 pin 8	PortL_2	LCD_COM2
6	LCD_COM3	LCD common 3 - J11 pin 7	PortL_3	LCD_COM3
7	LED1	LED1	PortR_0	GPIOR_0
8	LED2	LED2	PortR_1	GPIOR_1
9	LED3	LED3	PortR_2	GPIOR_2
10	LED4	LED4	PortR_3	GPIOR_3
11	LED5	LED5	PortR_4	GPIOR_4
12	LED6	LED6	PortR_5	GPIOR_5
13	LED7	LED7	PortR_6	GPIOR_6
14	LED8	LED8	PortR_7	GPIOR_7
15	+3.3V			
16	GND			

**Table 29: J27: Digital I/O**

Care should be taken to avoid trying to drive the port outputs high when the DIP switches are closed, as the closed switches short the port pins to GND.

The additional connections between the peripheral signals made on the daughter board are shown in the following diagram.



### 8.8.8 J25, J26: Enable LEDs

When the interconnection daughter board is fitted to headers J2 and J27, then headers J25 and J26 connect the eight user LEDs to the eCOG1X control signals. To connect the LEDs to the signals, link across all pins on headers J25 and J26 (1-2, 3-4, etc.). To disconnect the LEDs and leave these signals available for use elsewhere, remove the links. The two DIP switches are always connected.

Function	Port	J25 Pins		Name
GPIOR_0	PortR_0	1	2	LED1
GPIOR_1	PortR_1	3	4	LED2
GPIOR_2	PortR_2	5	6	LED3
GPIOR_3	PortR_3	7	8	LED4

Table 30: J25: LEDs 1-4

Function	Port	J26 Pins		Name
GPIOR_4	PortR_4	1	2	LED5
GPIOR_5	PortR_5	3	4	LED6
GPIOR_6	PortR_6	5	6	LED7
GPIOR_7	PortR_7	7	8	LED8

Table 31: J26: LEDs 5-8

### 8.8.9 J28: Piezo Sounder Configuration

Header J28 connects the piezo sounder input to either the PWM1 or the PWM2 output signal. Connect J28 pins 1-2 to select PWM1 (PortR\_0) to drive the sounder, and connect pins 2-3 to select PWM2 (PortR\_1).



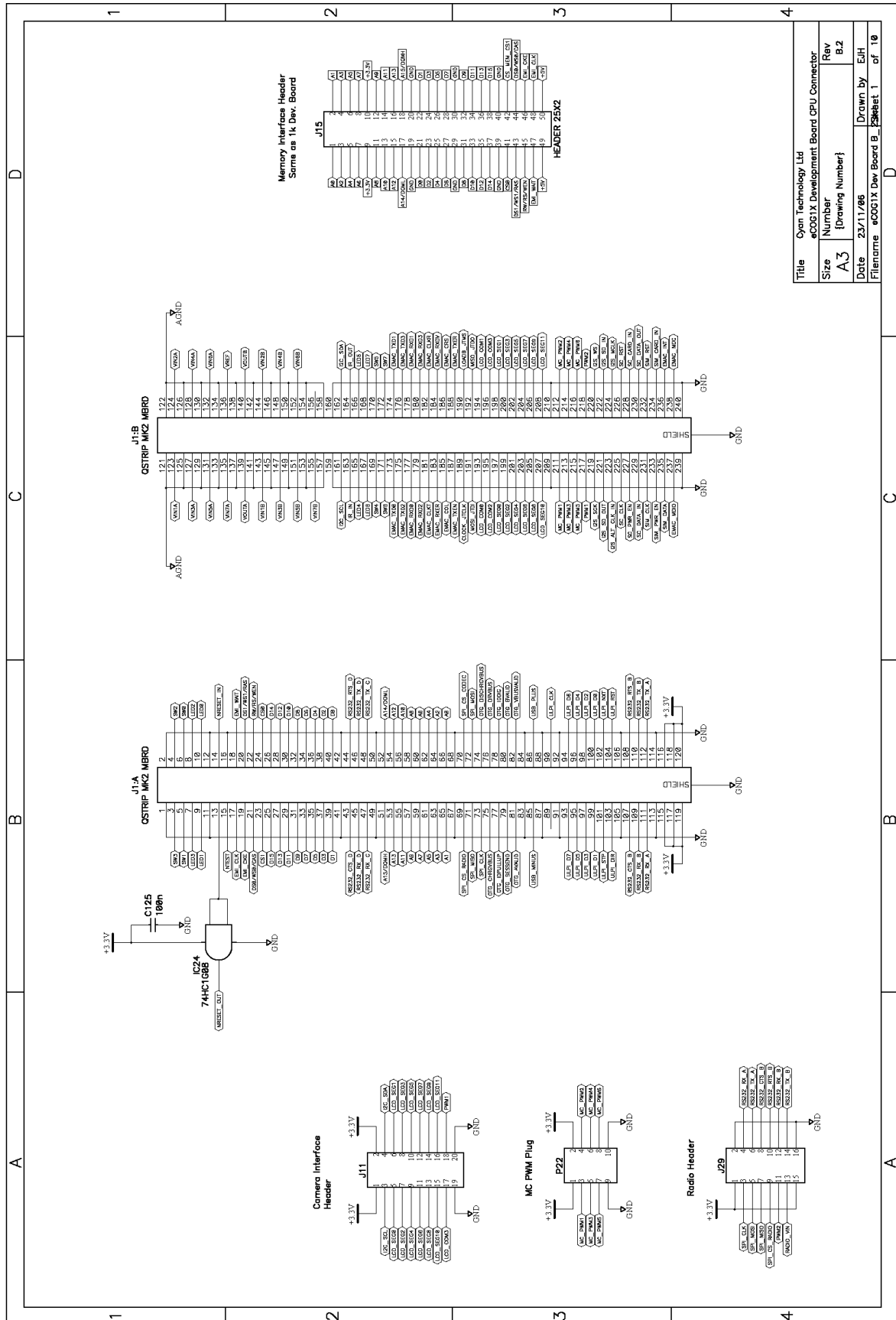
## Appendix A Important Notes

The following recommendations should be observed when using the USB eICE adaptor.

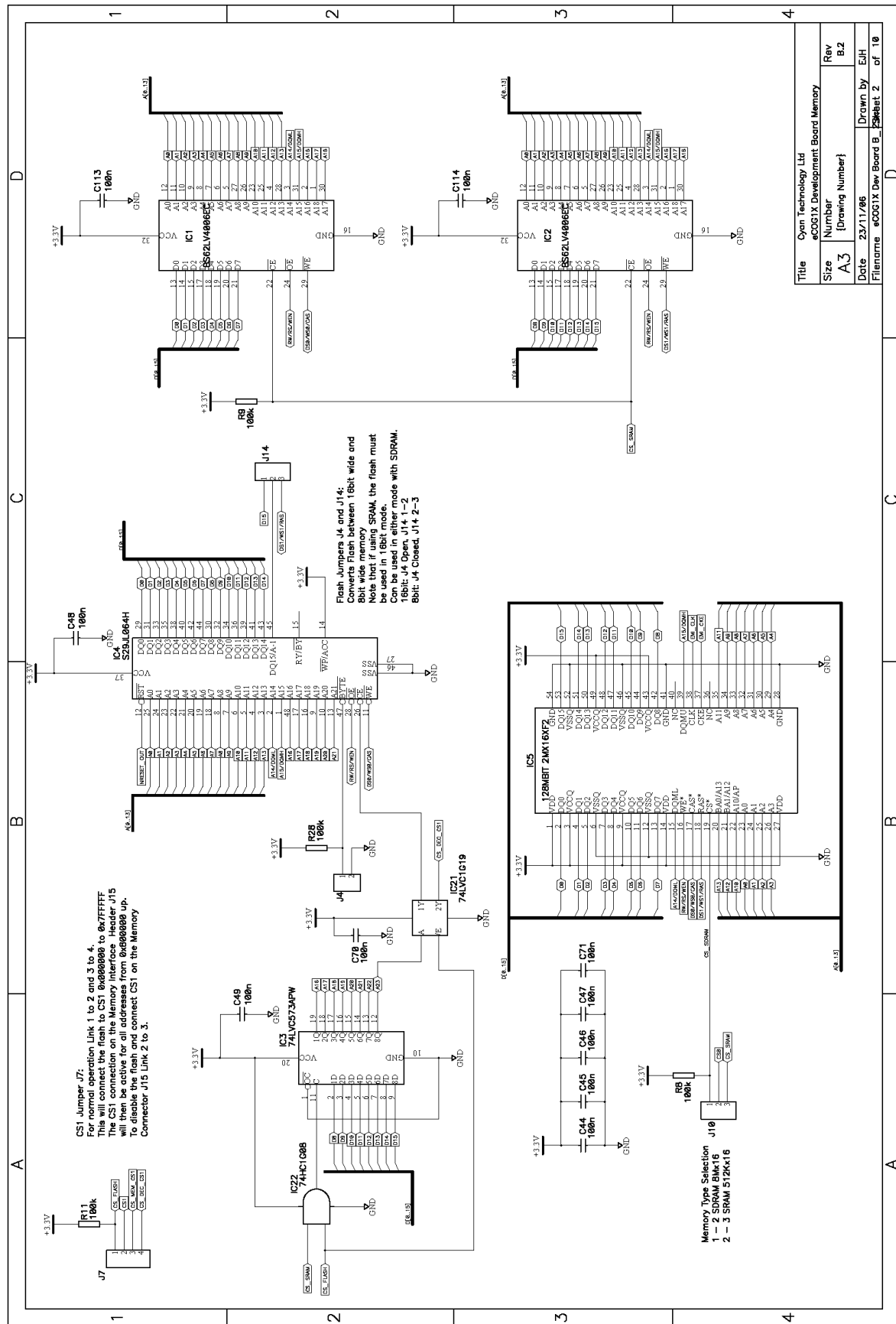
- Connect the USB cable from the eICE adaptor directly to the host PC, not via an external hub. CyanIDE can fail to restart the eICE debug connection after any errors if the USB device is connected via an external hub.
- The memory window in CyanIDE can be quite slow to refresh across the USB eICE link. Close the memory window when it is not required to improve the speed of response to commands.
- Do not disconnect the USB eICE cable or power down the target system while CyanIDE is running. This can cause CyanIDE to hang up on the next attempt to connect to the target system via eICE.  
Ensure that CyanIDE is closed down before disconnecting the USB cable or powering down the target system.
- CyanIDE may report an error message on its first attempt to connect to the target system via eICE. This occurs when it tries to find a connection to a target system on the parallel port instead of the USB port. Repeat the command and CyanIDE should connect to the USB eICE target system successfully.  
If the debugger still does not start, check that power is present on the target system, that all required jumper links are fitted, and that the USB cable is connected correctly to the host PC.

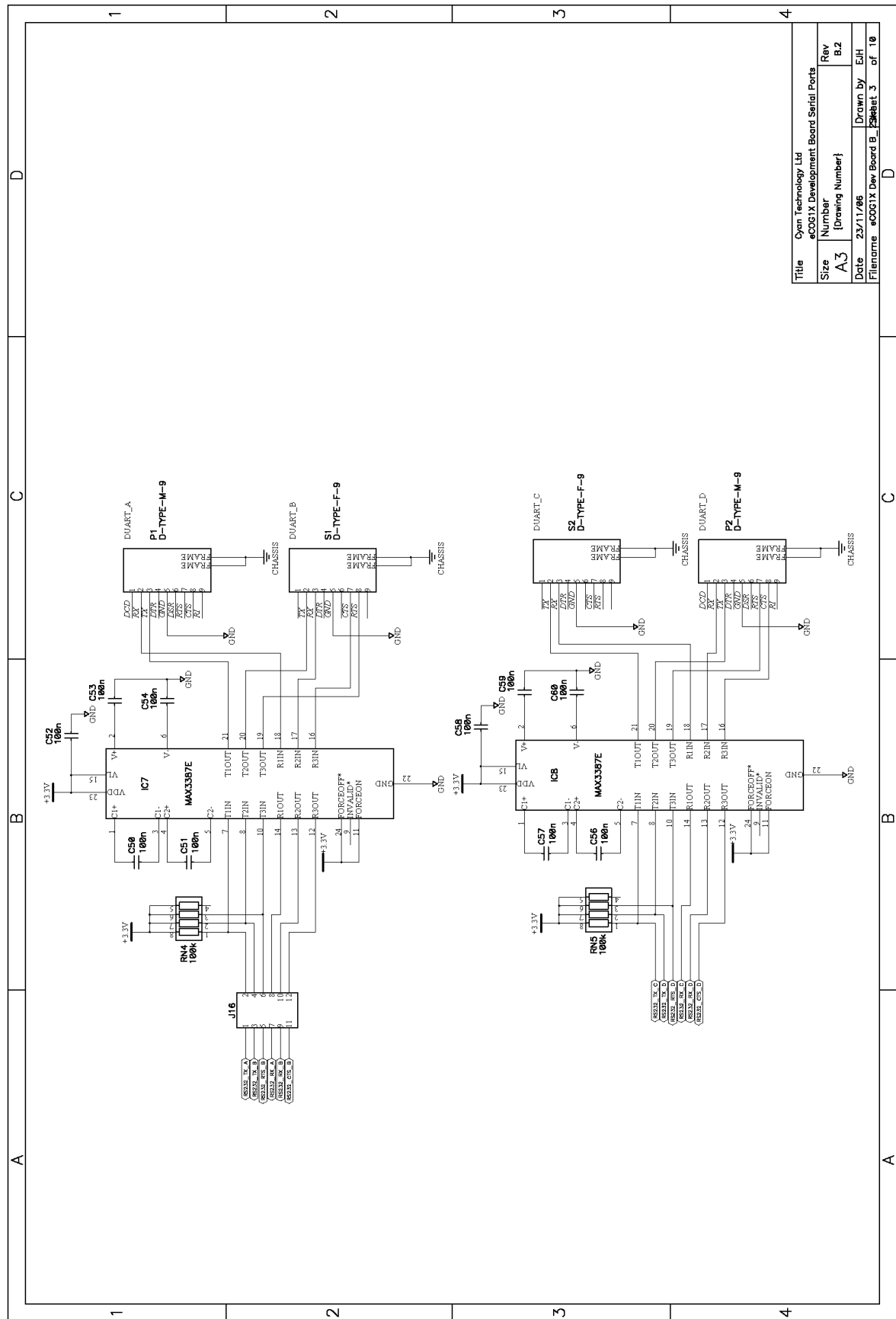
# Appendix B Circuit Diagrams

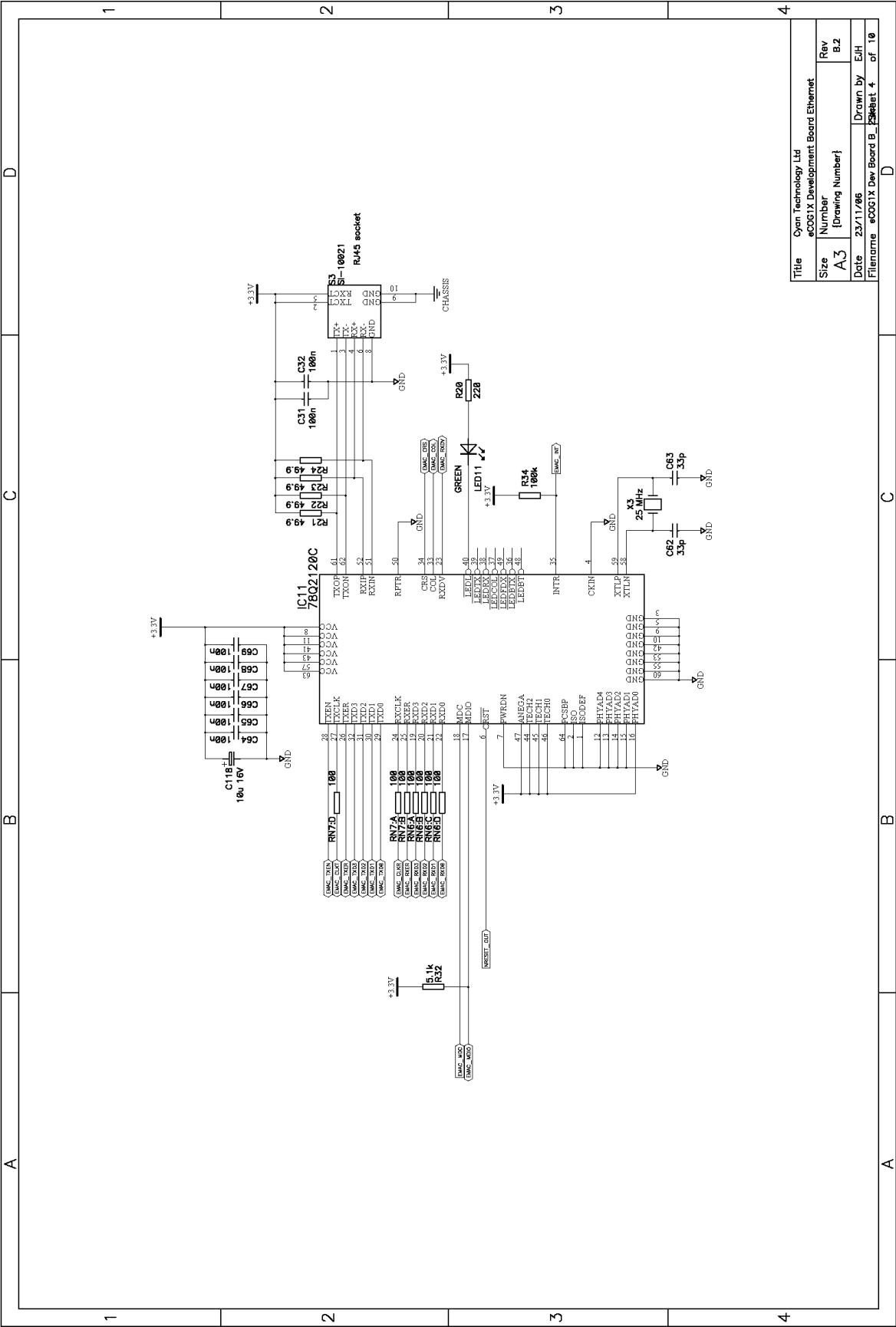
## B.1 Development Board

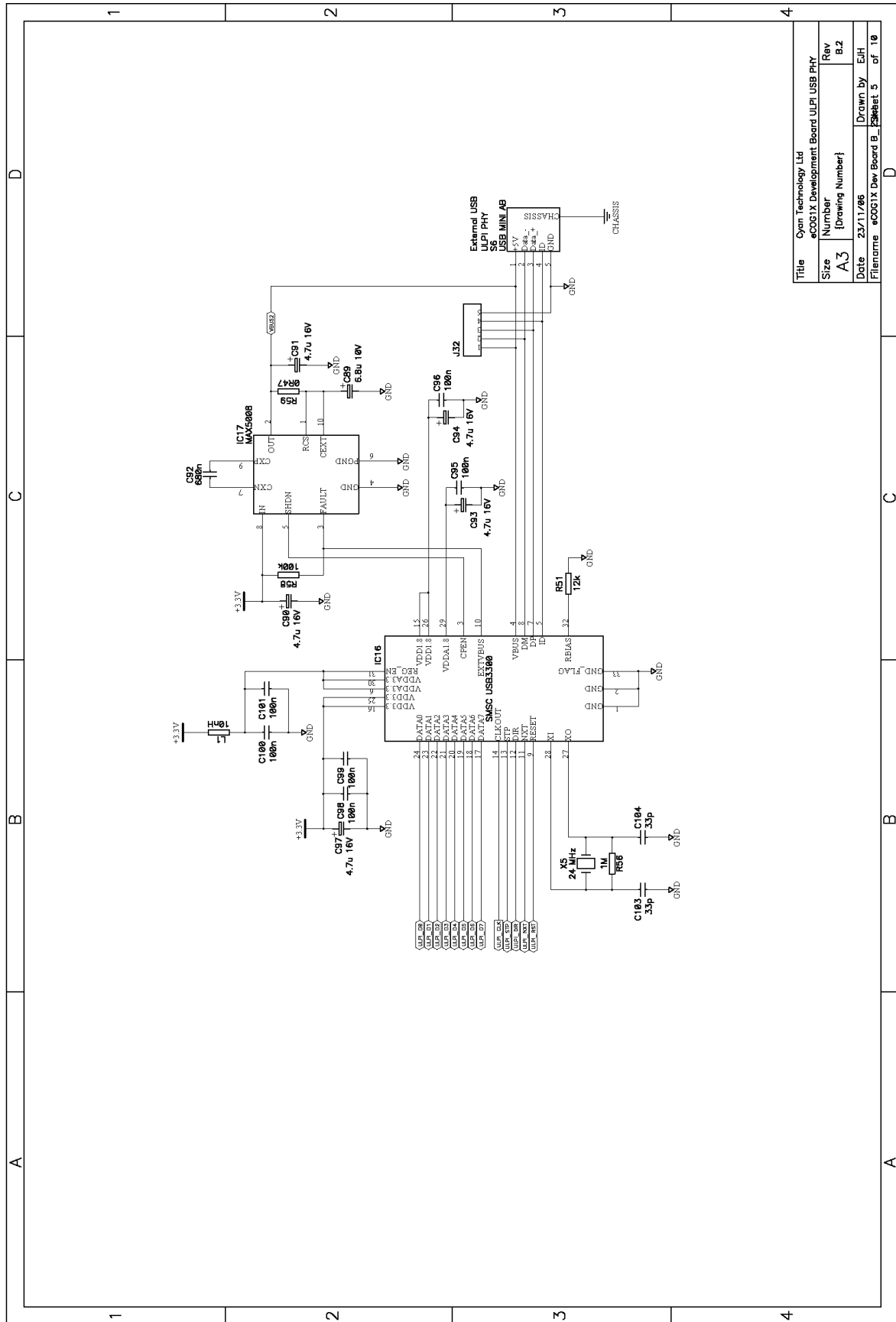


Title	Cyan Technology Ltd
Size	eCOG1X Development Board CPU Connector
Number	
Rev	B.2
Date	23/11/06
Drawn by	EJH
File name	eCOG1X Dev Board B.2
Sheet	1 of 10

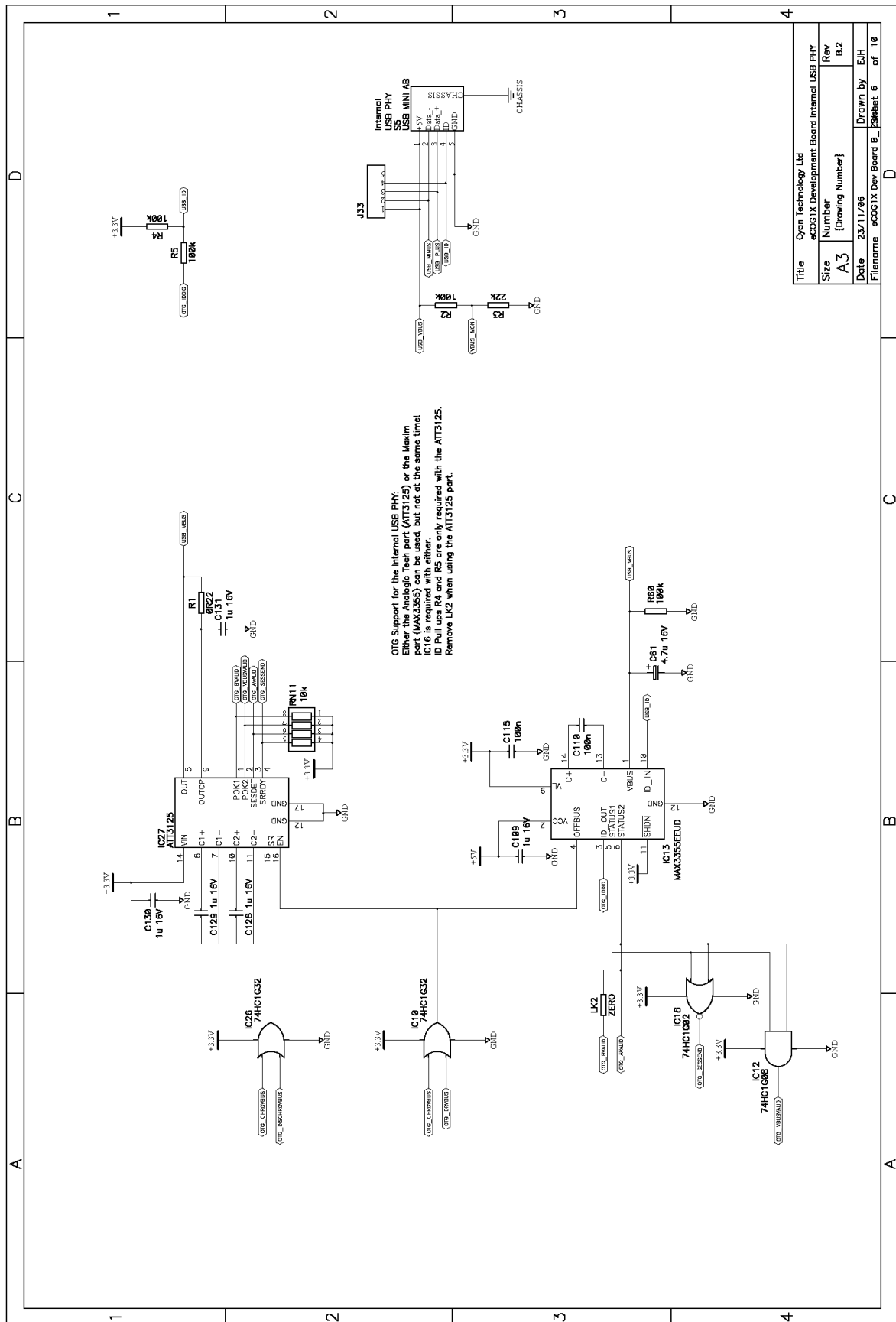


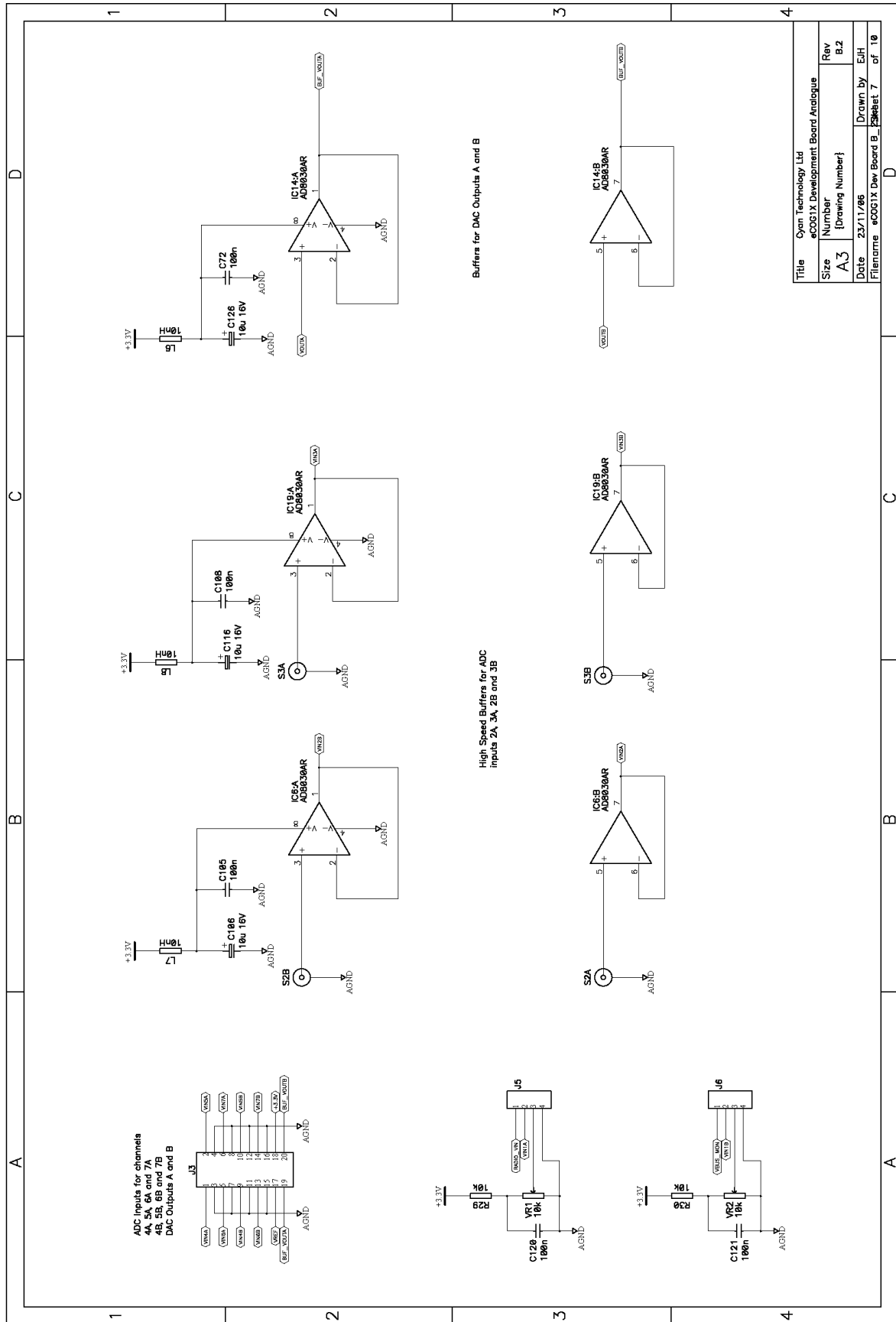




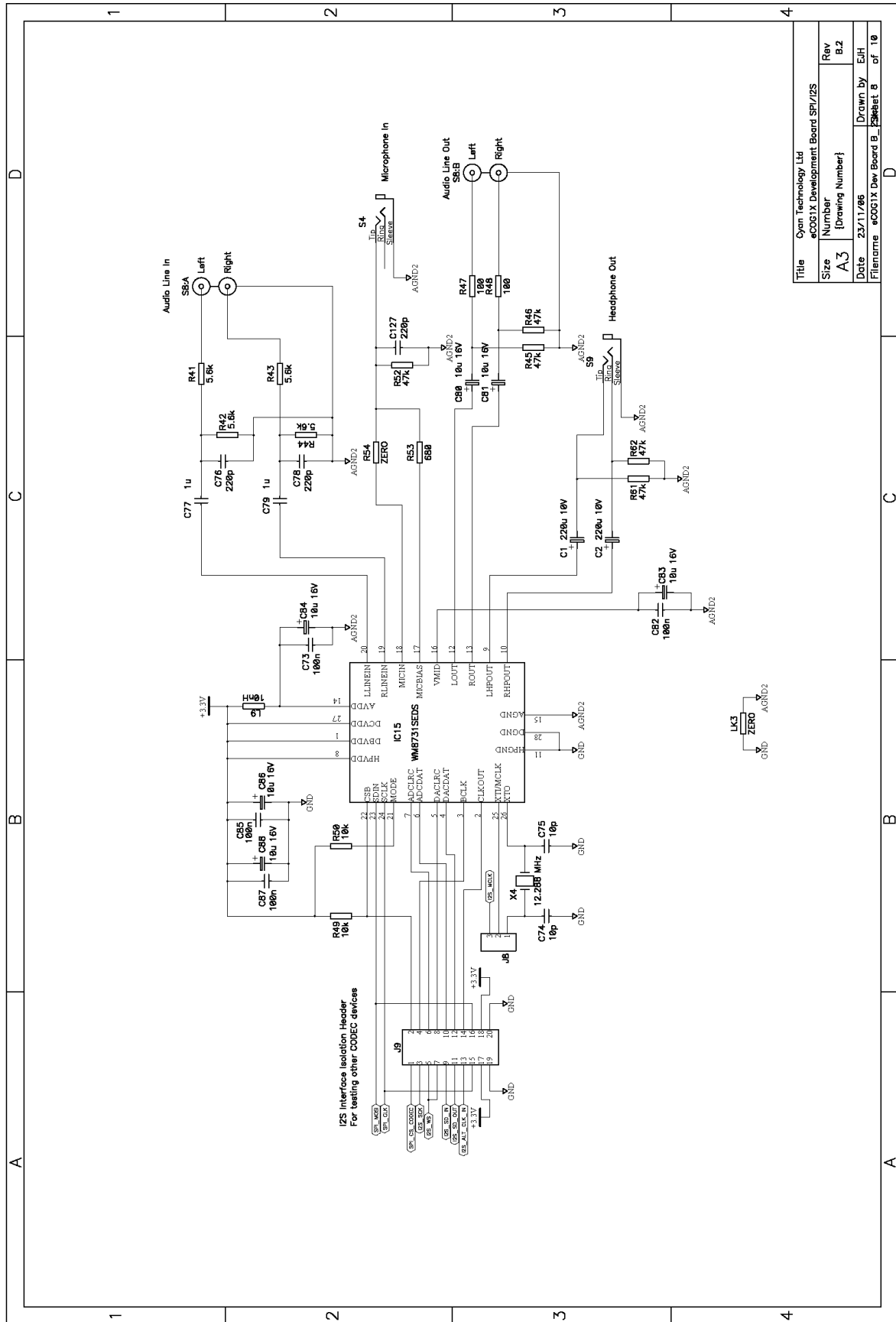


Title	Cyan Technology Ltd	eCOG1X Development Board UPLI USB PHY
Size	Number	Rev
A3	{Drawing Number}	B.2
Date	23/11/06	Drawn by E.H.
Filename	eCOG1X Dev Board B.	Sheet 5 of 10

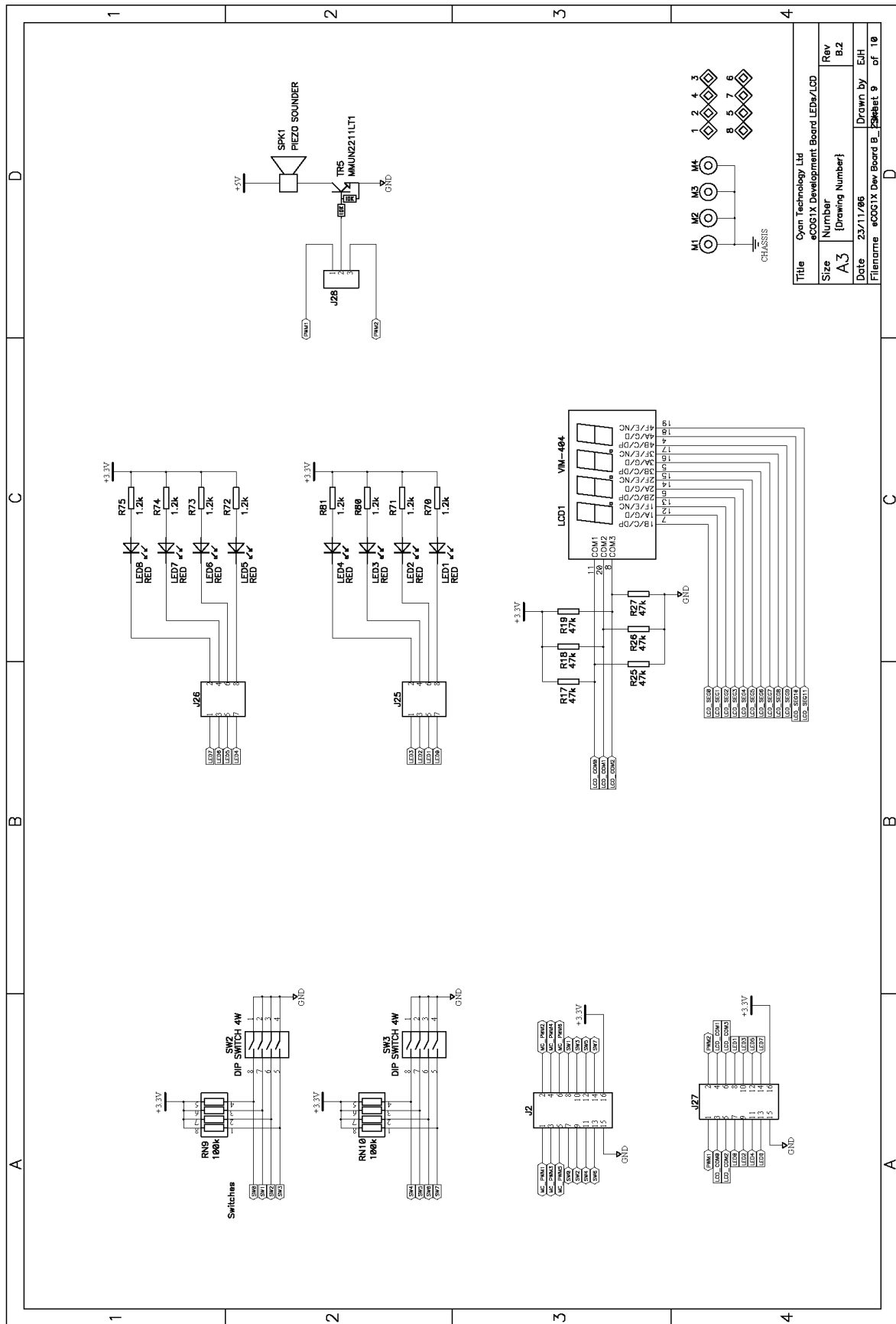




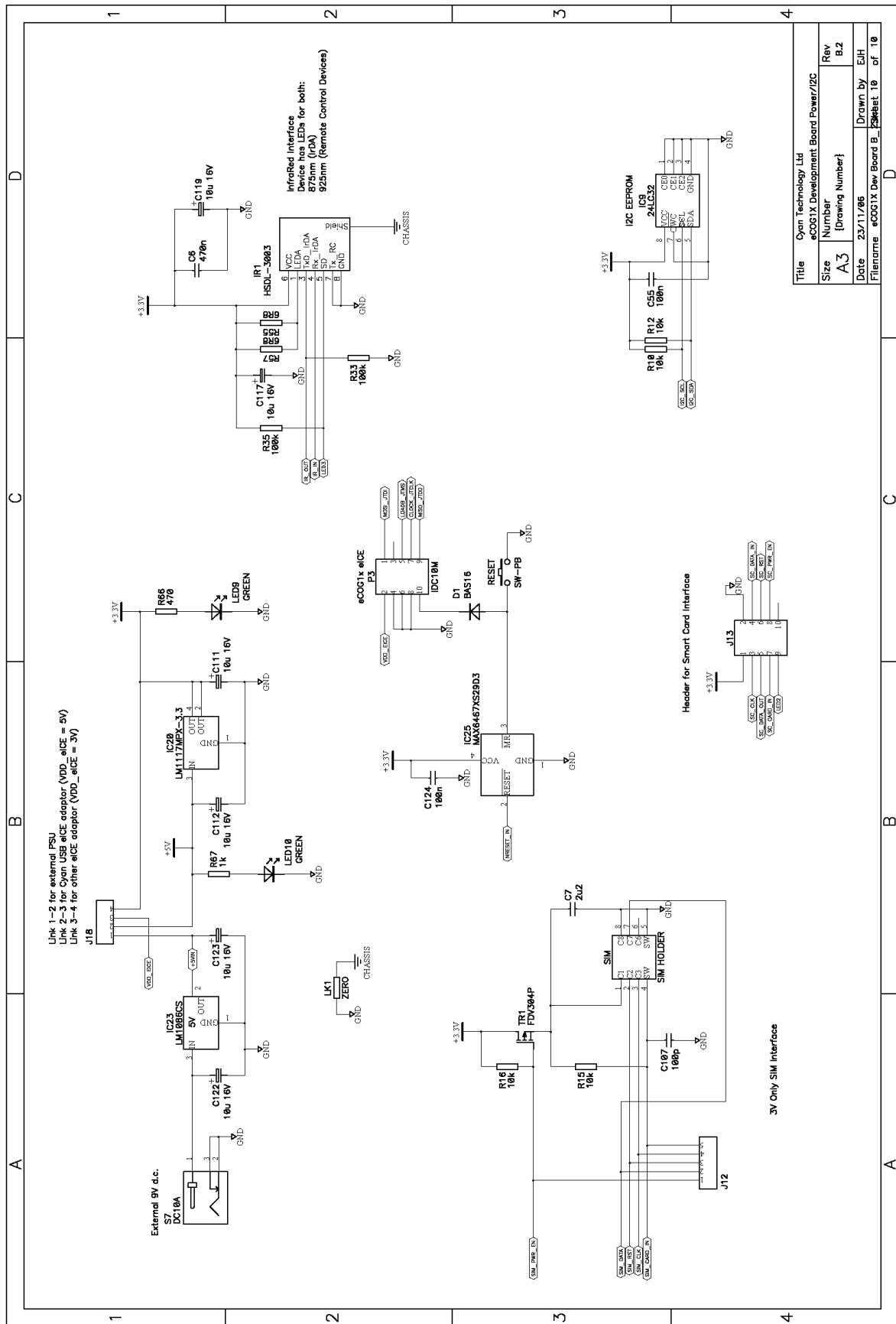




Title	Cyan Technology Ltd
Size	eCOG1X Development Board SP/12S
Number	Rev B.2
Drawing Number	A3
Date	23/11/06
Drawn by	EJH
Filename	eCOG1X Dev Board B. Sheet 8 of 10

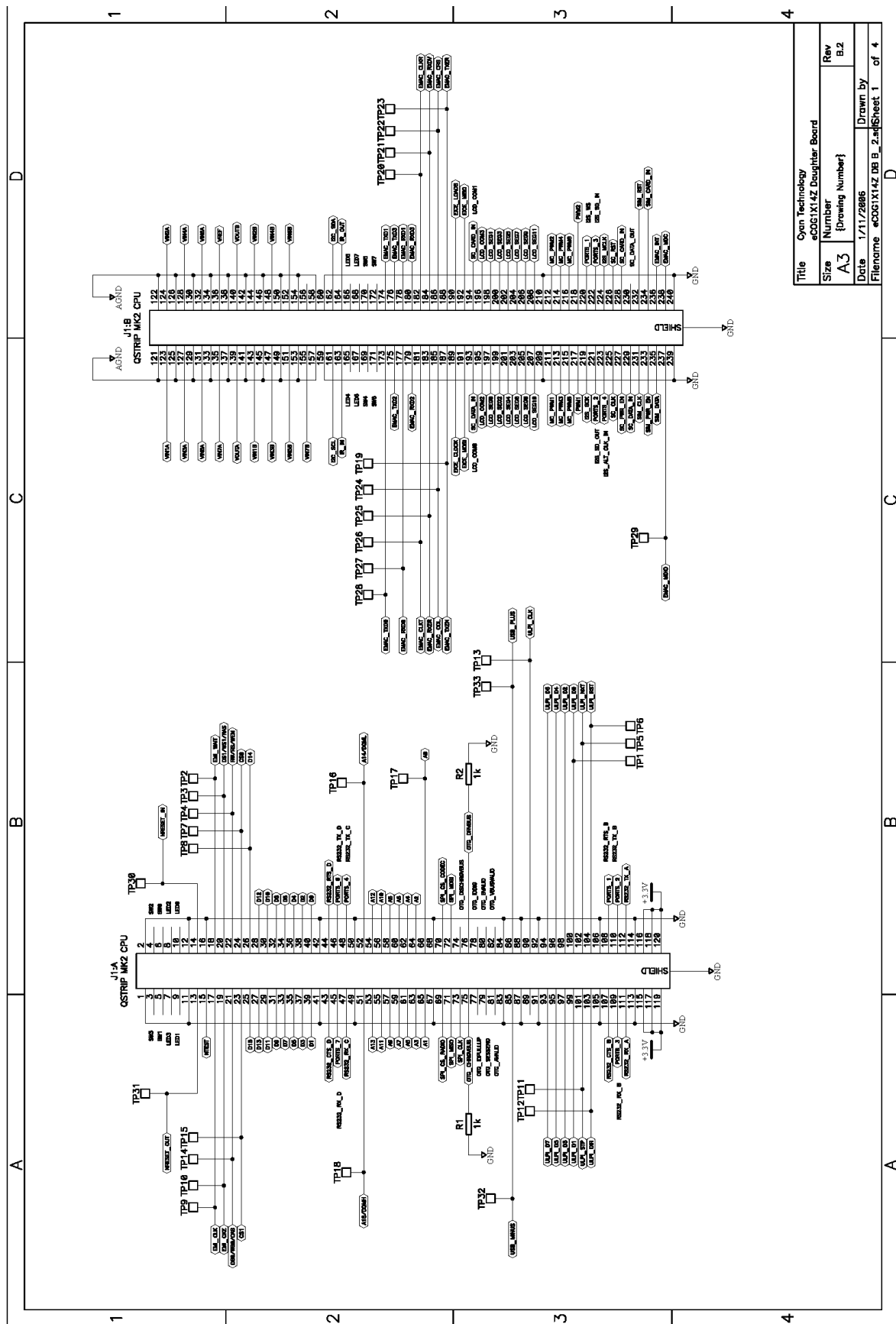


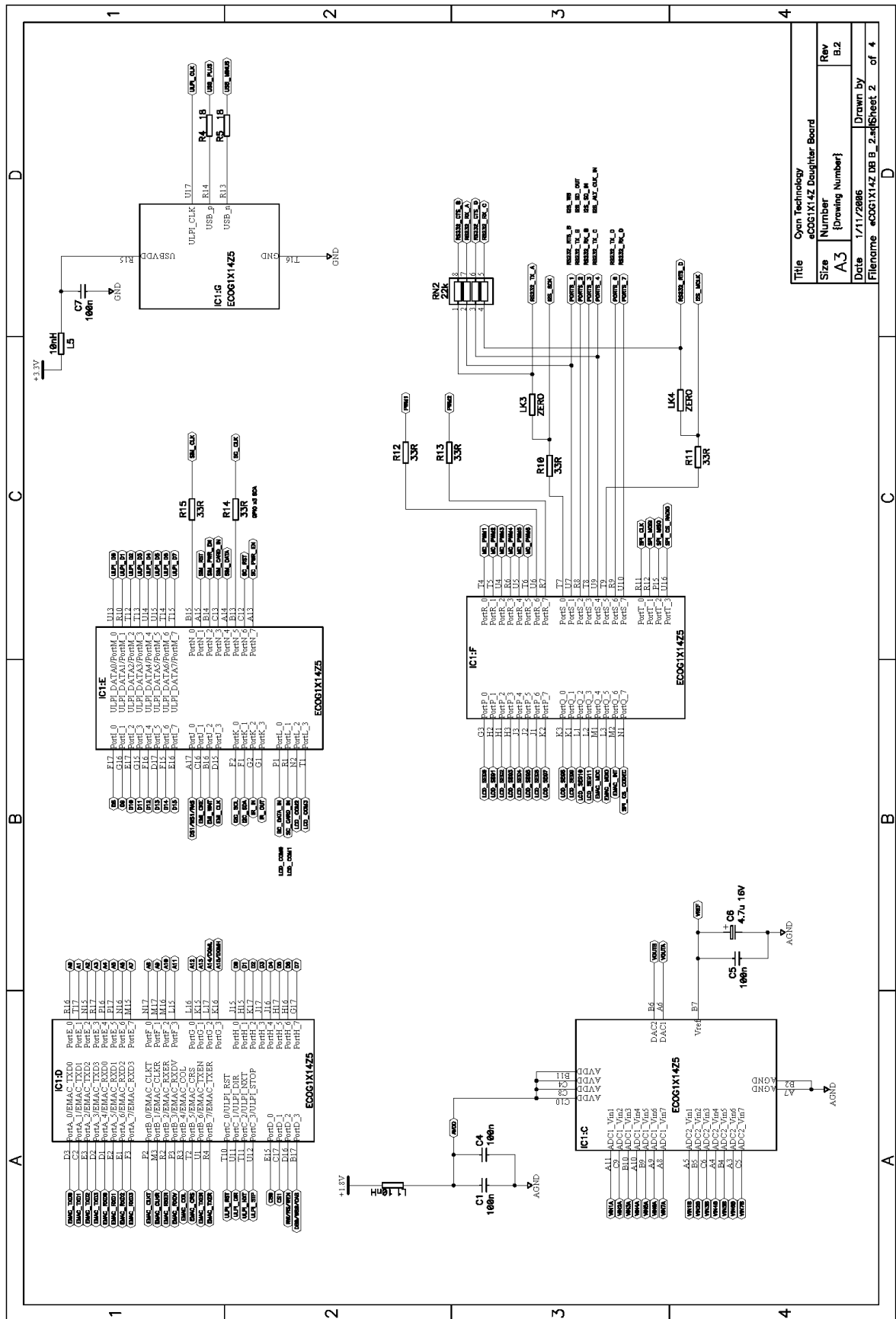
Title	Cyan Technology Ltd
Size	eCOG1X Development Board LEDs/LCD
Number	[Drawing Number]
Rev	B.2
Date	23/11/06
Filename	eCOG1X Dev Board B_Sheet 9 of 10
Drawn by	EJH

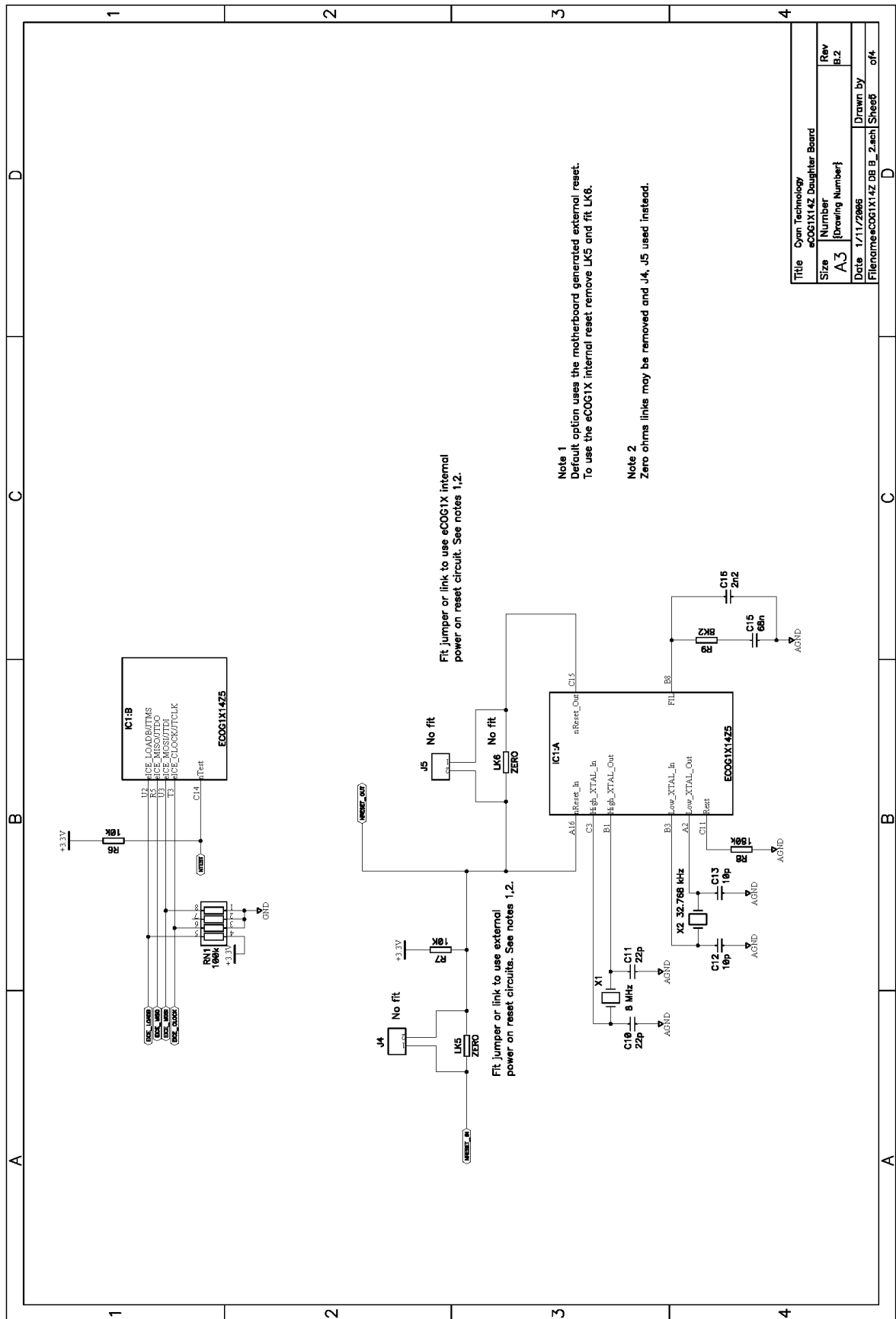


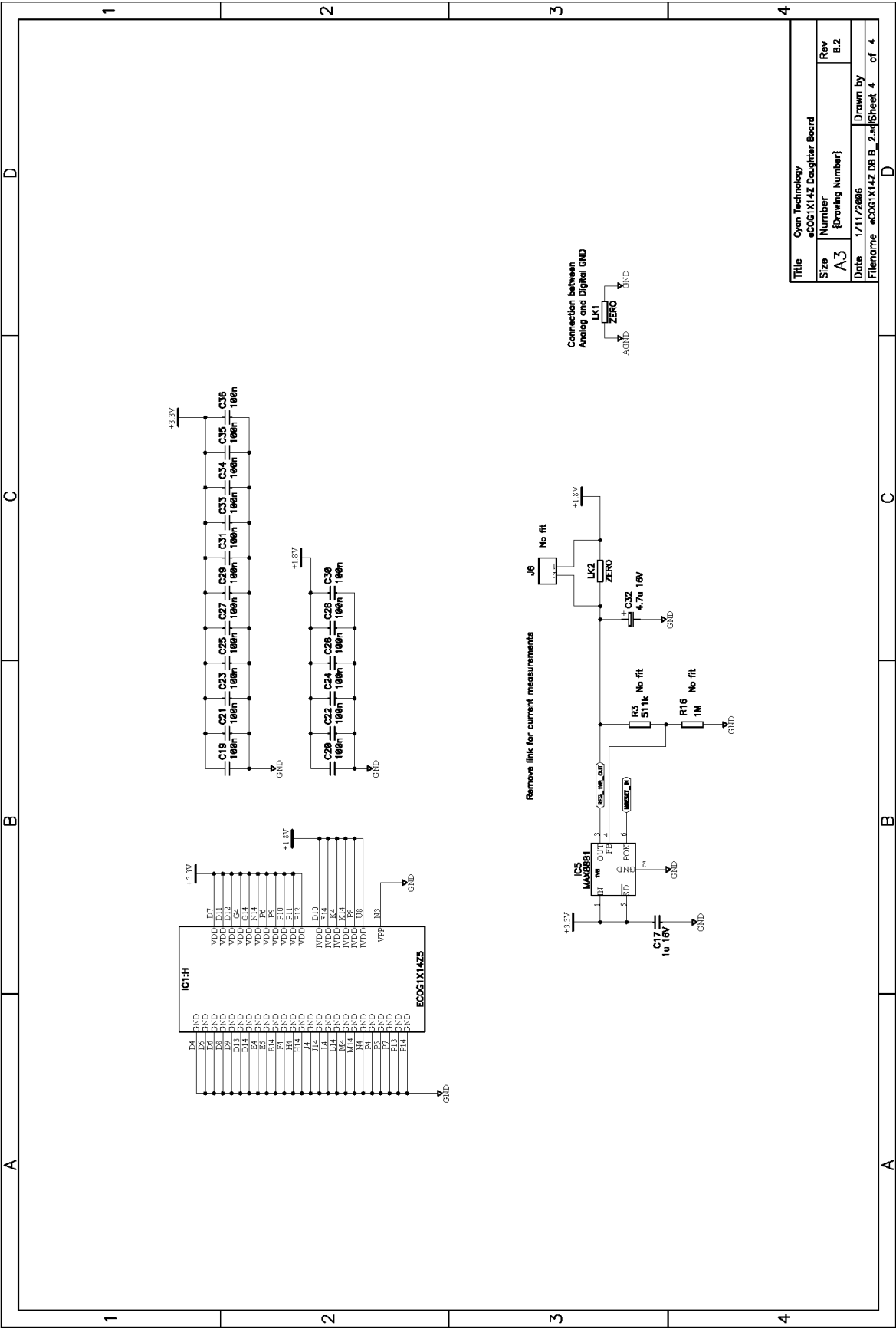
Title	Cyan Technology Ltd	eCOG1X Development Board Power/I2C
Size	Number	Rev
A3	{Drawing Number}	B.2
Date	23/11/06	Drawn by
Filename	eCOG1X Dev Board B.	Sheet 10 of 10

## B.2 Daughter Board (eCOG1X14Z5)



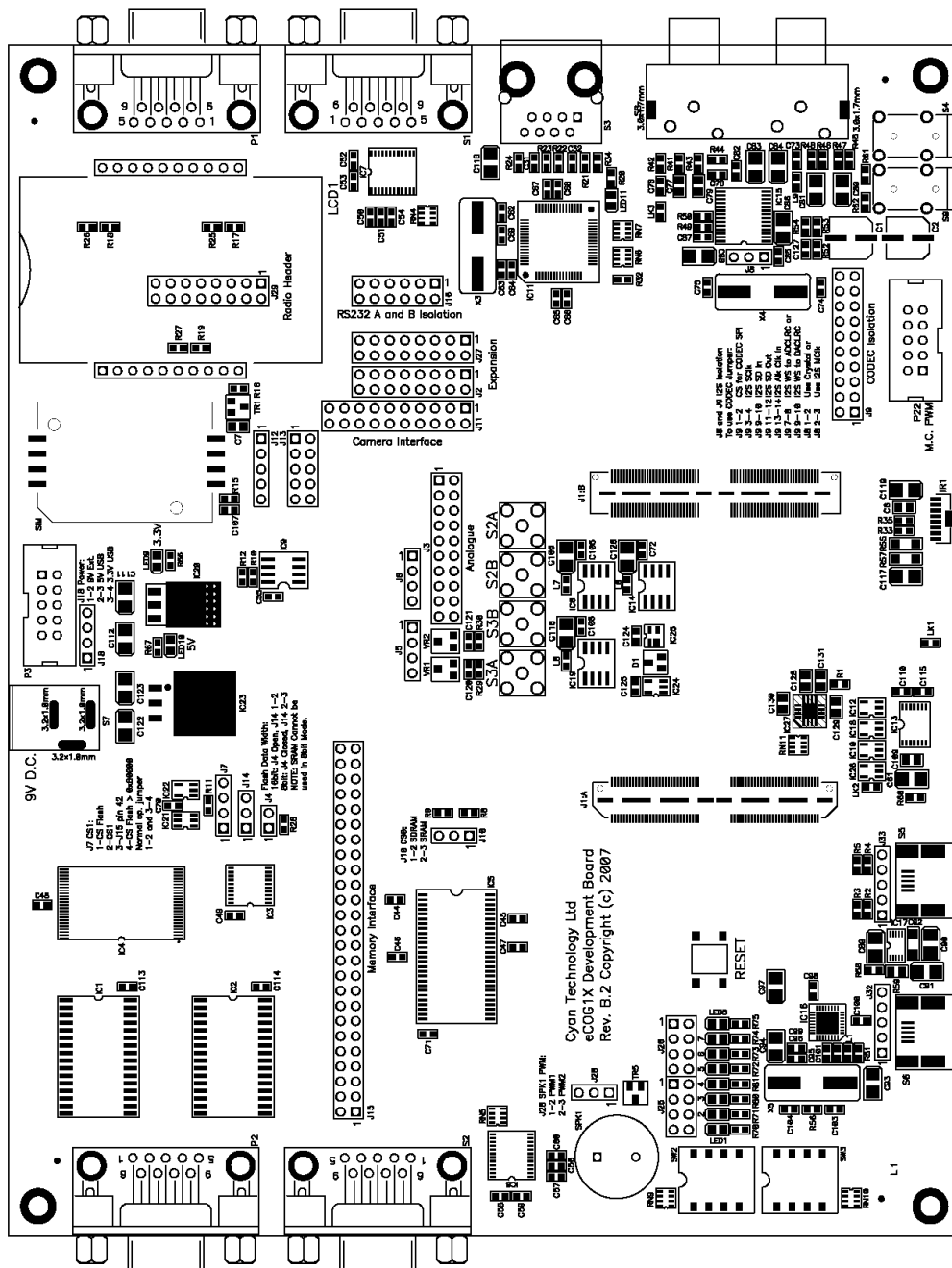






## Appendix C Board Layouts

## C.1 Development Board





## C.2 Daughter Board (eCOG1X14Z5)

